Aalto University
School of Science
Master's Programme in Computer, Communication and Information Sciences

Héctor Laria Mantecón

Deep Reinforcement Sequence Learning for Visual Captioning

Master's Thesis
Espoo, August 8, 2019

DRAFT! — August 8, 2019 — DRAFT!

Supervisor: Docent Jorma Laaksonen, Aalto University
Advisor: Docent Jorma Laaksonen, Aalto University

Aalto University
School of Science
Master's Programme in Computer, Communication and Information Sciences

ABSTRACT OF MASTER'S THESIS

Author:	Héctor Laria Mantecón		
Title:	Deep Reinforcement Sequence Learning for Visual Captioning		
Date:	August 8, 2019	**Pages:**	77
Major:	Machine Learning, Data Science and Artificial Intelligence	**Code:**	SCI3044
Supervisor:	Docent Jorma Laaksonen		
Advisor:	Docent Jorma Laaksonen		

Methods to describe an image or video with natural language, namely image and video captioning, have recently converged into an encoder-decoder architecture. The encoder here is a deep convolutional neural network (CNN) that learns a fixed-length representation of the input image, and the decoder is a recurrent neural network (RNN), initialised with this representation, that generates a description of the scene in natural language.

Traditional training mechanisms for this architecture usually optimise models using cross-entropy loss, which experiences two major problems. First, it inherently presents exposure bias (the model is only exposed to real descriptions, not to its own words), causing an incremental error in test time. Second, the ultimate objective is not directly optimised because the scoring metrics cannot be used in the procedure, as they are non-differentiable. New applications of reinforcement learning algorithms, such as self-critical training, overcome the exposure bias, while directly optimising non-differentiable sequence-based test metrics.

This thesis reviews and analyses the performance of these different optimisation algorithms. Experiments on self-critic loss denote the importance of robust metrics against gaming to be used as the reward for the model, otherwise the qualitative performance is completely undermined. Sorting that out, the results do not reflect a huge quality improvement, but rather the expressiveness worsens and the vocabulary moves closer to what the reference uses.

Subsequent experiments with a greatly improved encoder result in a marginal enhancing of the overall results, suggesting that the policy obtained is shown to be heavily constrained by the decoder language model. The thesis concludes that further analysis with higher capacity language models needs to be performed.

Keywords:	deep learning, machine learning, neural networks, reinforcement learning, policy gradient, reinforce, self critic, captioning, description generation, computer vision
Language:	English

Acknowledgements

Firstly, I would like to thank my supervisor Jorma Laaksonen for opening his door and giving me a vote of confidence to join his group and projects. Thanks for being patient while I was learning the basics and about myself. And of course, for his good guidance during the thesis.

I would like to thank my parents for giving me the opportunity to reach this point, and to them and my partner for supporting and taking care of me when I do not. Thanks to my friends here and those who stay at home, for encouraging me through rough periods and helping me to take a mental or physical break from the stresses of life when most needed.

I would like to thank Ricardo Falcón for his guidance and feedback during the writing of this thesis and the masters in general, and to Ismail Harrando for his feedback on early drafts too. I am also grateful to everyone in the CBIR group, especially to my office colleagues for fruitful and inspiring (although scarce!) discussions.

Last but definitely not least, such an endeavour could be achieved thanks to all my masters' friends and fellow classmates. Thanks for the group work, reciprocal support and positive attitude during the most exasperating times, chiefly to Maximilian Proll, who patiently filled gaps on my math skills and bore with my snail's pace.

Thank you, I would not be the same person today without any of you.

Espoo, August 8, 2019

Héctor Laria Mantecón

Abbreviations and Acronyms

COCO	Microsoft Common Objects in Context dataset
EOS	End of sequence
I3D	Inflated 3D convolutional network
LSTM	Long Short-Term Memory Network
ML	Machine Learning
MT	Machine Translation
NLP	Natural Language Processing
NN	Neural Network
ResNet	Residual Network
RNN	Recurrent Neural Network
SGD	Stochastic Gradient Descend
SOS	Start of sequence
TGIF	tumblr gif dataset
TREC	Text REtrieval Conference
TRECVID	TREC Video Retrieval Evaluation
VTT	Video to text

Contents

Chapter 1

Introduction

In a society that relies heavily on audiovisual content as a source of information and entertainment, an inherent risk of exclusion arises for people with barriers to the access to this content, such as visual impairment. To reduce these barriers, certain technologies have been developed in recent years. The ability to automatically describe image or video data using natural English sentences is called *image* or *video captioning*. Some applications include lowering the barrier for people to information sources, improving understanding of web content for blind people, or generating meta-data on media sources.

The captioning task is hardly a challenge for humans due to our remarkable ability to assimilate and compress enormous amounts of visual information, which can be later transmitted by descriptive language. However, massive generation of human descriptions rapidly becomes a costly exercise, since an annotator can only work sequentially and needs resting periods after a short time. Some efforts exist, such as Amazon Mechanical Turk [9], that provide a platform for parallel manual tasks, but it is an arduous job and can easily escalate to prohibitive costs.

On the contrary, automatic description presents a great challenge for machine learning algorithms and is harder than well-studied image classification or object recognition tasks. Concretely, it requires an acute understanding of local and global entities, along with their respective attributes, relationships and activities involved. This knowledge has to additionally be expressed in a properly formed human language such as English, thus a language model is needed on top of it. Nevertheless, large-scale automatic description implies virtually no monetary cost and can be performed continuously and in parallel.

Several works such as MS COCO [11] provide extensive datasets and foster research by enabling evaluation systems and online benchmarking for competing methods to come. Since their inception, deep learning approaches

on sequence modelling have dominated the leaderboard, owing to their improved quality of caption generation. This quality is determined by advances in neural network training [30] and the publication of better classification datasets [50], as well as progress in the neighbouring research field of machine translation (MT) [12].

MT improvements started when instead of word translation, alignment and reordering, Recurrent Neural Networks (RNNs) were applied to read the source sentence with an encoder RNN to gain a fixed-length representation of it, followed by a decoder RNN initialised with this representation that generates the target sentence. Efforts on visual captioning drawing upon these successes [60] resulted in replacing the encoder RNN by a deep convolutional neural network (CNN) that would learn a fixed-length representation of the input image. This representation would be fed to the decoder RNN to produce descriptions.

This new design is not without its problems, as the model suffers from *exposure bias* [47] (the model is only exposed to real descriptions, not to its own words) while being trained using the general Teacher-Forcing algorithm [6]. This algorithm usually optimises models using the cross-entropy loss, while the model performance at the test time is assessed with discrete and non-differentiable metrics such as BLEU [43] or CIDEr [58]. Fortunately, it has been shown [47] that the use of REINFORCE algorithm [63] from Reinforcement Learning overcomes the exposure bias issue, while directly optimising non-differentiable sequence-based test metrics therefore achieving a new state-of-the-art benchmark score.

1.1 Problem statement

When captioning models, and more concretely their language submodels, are trained for sequence output, the Teacher-Forcing algorithm [6] is normally used. This algorithm inherently presents exposure bias [47], causing an incremental error at the test time. Some methods such as Professor-Forcing [33] have been designed to mitigate this issue. Additionally, the training metric used is cross-entropy (to be discussed in Section 3.2), while the test metric that ultimately matters is a natural language processing metric (to be discussed in Section 3.5). Intuitively, one would use the last metric as the optimisation objective, but unfortunately, these metrics are non-differentiable so they cannot be used to compute gradients.

Nevertheless, recent reinforcement learning applications in sequence training [48] have managed to incorporate these scoring functions into differentiable methods to drive the model training, while getting rid of the exposure

bias as well. Moreover, they do not suffer from the shortcomings of reinforcement learning such as estimating the reward signal or the normalisation to be applied.

The main focus of this thesis is to review and analyse the performance of these different captioning methods, identifying strong and weak points, studying the mechanics and comparing the output quality. The same model architecture and configurations are going to be applied to have a baseline and a fair comparison. The thesis also discusses how rethinking and reformulating a problem from a different perspective can bring improvements over a state of plateau performance.

1.2 Structure of the thesis

Chapter 2 provides a historical perspective of captioning models until current, state-of-the-art architectures, followed by a review of policy gradient methods for sequence training in the area of reinforcement learning. Lastly, descriptions of relevant datasets used for training and testing are presented.

Chapter 3 describes the methodology used for the experiments. The model to be used and how the research has led to it are presented, followed by how the model actually learns, by detailing the training losses to be optimised. After that, performing classification models for image and video data are showcased, which are used for feature extraction for the description generation. The chapter concludes by defining sequence scoring metrics widely used in Natural Language Processing applications and therefore also used in this work to score the results.

In Chapter 4, the setup for all experiments is motivated and detailed. First, it is denoted the configuration of the datasets for each training, validation and testing stages. Second, the feature extraction procedure is explained for each media modality, including specific values and final feature combinations. Next, the characteristics of the vocabulary used by the models are clarified. The chapter finishes describing specific parameter settings from the code and pointing where the project implementation can be found.

Chapter 5 reflects the experiments performed in this thesis. It starts delineating a hyperparameter search followed by its results. Then the motivation and setup of each experiment is shown, along with an adequate characterisation of the results by the training dynamics, scores and output samples. It finalises displaying quantitative and qualitative analyses of the outputs, as well as reflections that can be extracted from the final results.

In Chapter 6, the results of the previous chapter are discussed and interesting findings are noted. Lastly, Chapter 7 concludes with a summarisation

of the findings of this thesis and final thoughts on future research are shared.

Chapter 2

Background

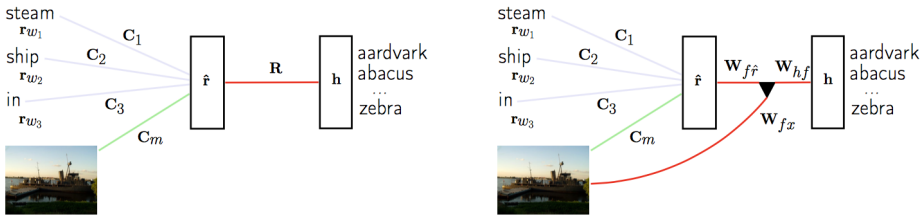
Visual captioning is the area of computer vision that generates natural language descriptions given images or videos. It is a difficult task, since it lays on the intersection between vision, for scene recognition, and language, for scene explanation. For the former, it requires performing detectors with the sensibility to capture as much information as possible. For the latter, high-capacity language models are required to express detections in such a nuanced sequential way as the human.

This chapter provides theoretical background knowledge for this thesis, by presenting the most relevant developments in captioning since its inception and explaining its motivations and challenges. Next, a relatively new branch of reinforcement learning application for sequence training is shown, which is applied to language generation to reach higher levels of refinement than traditional methods. Finally, the importance of data and particular datasets for captioning are addressed.

2.1 Visual captioning

Natural language description for visual data has been a long-running problem in computer vision [16], both for image and video data. The first models started with primitive recognisers followed by formal language generators [65], such as And-Or networks [1] or logic systems that follow rule-based methods [8] to generate natural language. These language systems were heavily hand-crafted and not really applicable to real-world data.

Relevant progress was made in image recognition, where objects, attributes, relationships and locations were detected and used to produce natural descriptions via template-based text generation [14, 35]. More complex detection graphs appeared [32] and better language systems were applied



(a) Modality-Biased Log-Bilinear Model (MLBL-B)

(b) Factored 3-way Log-Bilinear Model (MLBL-F)

Figure 2.1: Left: The predicted next word representation $\hat{r}$ is a linear prediction of word features $r_{w_1}, r_{w_2}, r_{w_3}$ ($C_{1...3}$ connections) biased by image features x. Right: The word representation matrix R is replaced by a factored tensor for which the hidden-to-output connections are gated by x. Figure from [28].

Figure 2.2: Multimodal log-bilinear descriptions, initialised with either "*in this picture there is*" or "*this product contains a*". The captions include some concepts from the pictures, but they often miss or misrecognise most of them, as well as adding generic gibberish. Figure from [28].

until the appearance of superior neural-based models.

One of the first neural network applications on this task was Multimodal neural language models [28], a multimodal log-bilinear model [40] conditioned on image features. The architecture is depicted in Figure 2.1 and generated samples can be found in Figure 2.2. This technique was followed by a similar approach [38] in the generation yet modifying the language model, by switching from feed-forward to recurrent layers. Long short-term memory (LSTM) recurrent networks (Section 3.1) were first used by [60] in their model, with the difference that the image was only shown to the RNN at the beginning of the description generation. Follow-up works [13] integrated LSTMs for video description too.

These recent methods based on Recurrent Neural Networks (RNNs) were inspired by the success of sequence to sequence (seq2seq) training used on

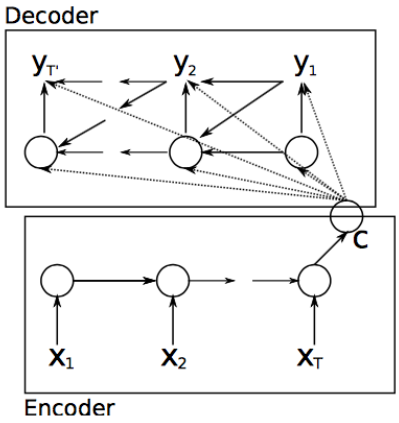


Figure 2.3: Encoder-decoder architecture. Input $\boldsymbol{x} = \{x_1, \ldots, x_T\}$ gets encoded into the fixed-length context vector $\boldsymbol{c}$, which is then used to produce the output $\boldsymbol{y} = \{y_1, \ldots, y_{T'}\}$ at each step. Figure from [12].

neural machine translation [5, 12]. The appearance of the encoder-decoder architecture from [12], depicted in Figure 2.3, translates well to image description because one can argue that the task boils down to "translating" an image to a sentence.

It was discussed in [61] that unidirectional, shallow LSTMs cannot generate contextually well-formed captions. In order to address this shortcoming, a bidirectional LSTM architecture was proposed. Using this biLSTM, the language model is able to utilise past and future context information to produce long-term language relations, which eventually leads to contextually and semantically richer captions, as seen in Figure 2.4.

There have also been efforts [25, 29] to learn a joint embedding space that would allow not only description, but also ranking of different descriptions, in the way of scoring captions and visual similarity. The first work [29] uses a log-bilinear model for the generation, which precises a fixed window context. The second [25] learns the embedding using a feature extractor and a bidirectional RNN, which conditions the generation on all the previous words produced, an improvement from the fixed window. This schema is shown in Figure 2.5.

Further related works [64] involved the use of attention, given its success in machine translation [5] and object recognition [3]. Figure 2.6 displays a diagram of the architecture. Here visual attention is applied on the encoder, to focus on salient parts of an image on the moment of the generation, as presented in Figure 2.7. A beneficial side effect is that it can be intuitively seen where the model "looks" while describing the scene, which can easily

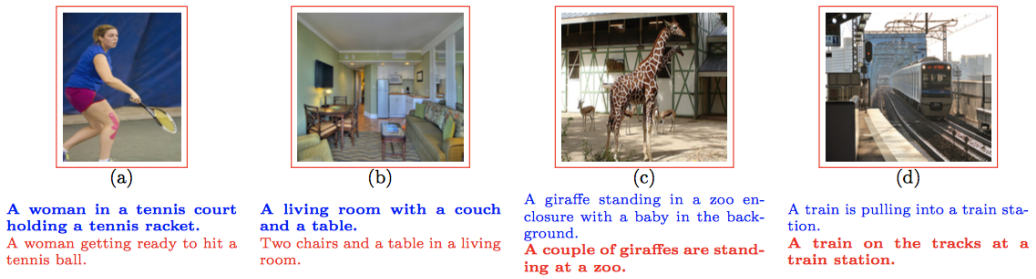


Figure 2.4: Examples of generated captions with a biLSTM for given query image on MS COCO (Section 2.3.1) validation set. The blue-coloured captions are generated in forward direction and the red-coloured captions are generated in backward direction. The final caption is selected according to the higher probability of the sentences and is marked in bold. Figure from [61].

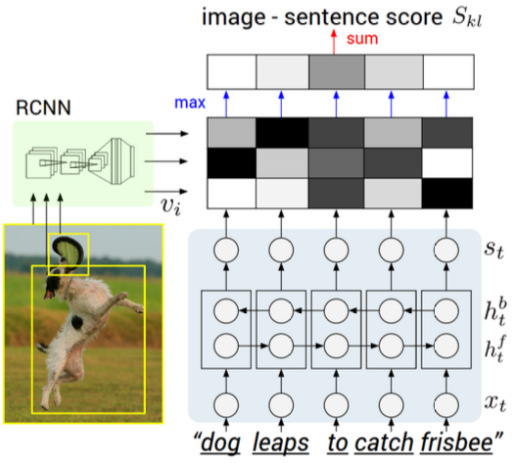


Figure 2.5: Object regions are embedded with a CNN (left). Words (enriched by their context) are embedded in the same multimodal space with a biRNN (right). Pairwise similarities are computed with inner products (magnitudes shown in grayscale). Figure from [25].

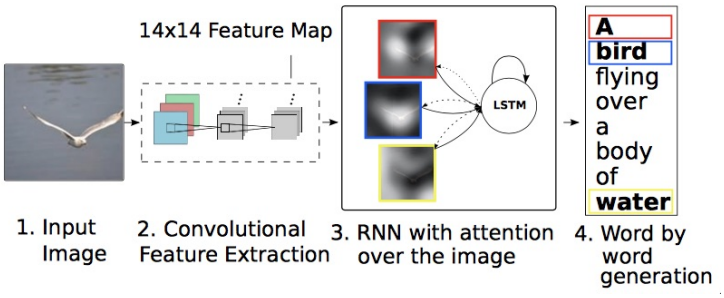


Figure 2.6: Visual attention and RNN model diagram. The model learns a word-image alignment. Figure from [64].

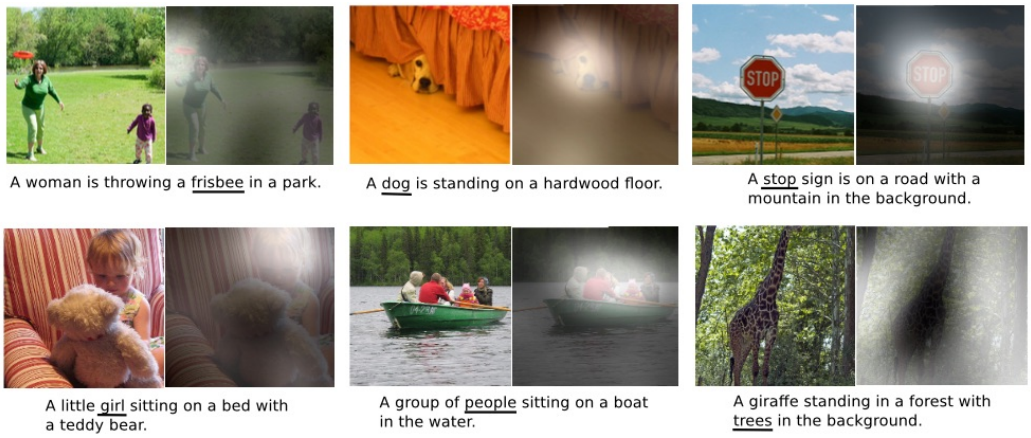


Figure 2.7: Generated captions of a model with visual attention, attending to the correct object (white indicates the attended regions, underlines indicate the corresponding word). Figure from [64].

point out flaws of a model.

The Transformer architecture [57], depicted in Figure 2.8, led to significant improvement in translation and other end-to-end language tasks. Its main benefit was to get rid of the recurrent dependency of RNNs and to rely solely on the attention applied to the encoder and decoder, which enabled them to be trained in parallel. This reported better performance than RNNs and faster training times. Drawing upon machine translation again, this method has been recently applied [51] as both encoder and language model for image captioning with positive results. A caption comparison can be seen in Figure 2.9.

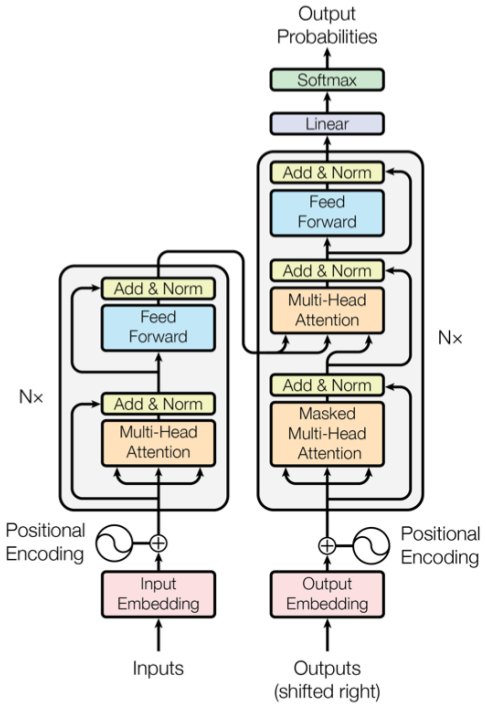


Figure 2.8: Transformer architecture. Figure from [57].

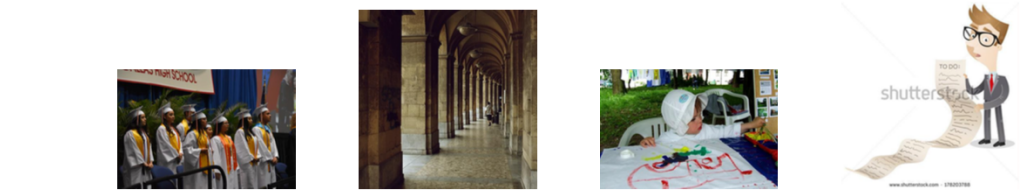


Figure 2.9: Captions comparison on a recurrent neural network (RNN8x8) and a transformer (T2T8x8) models trained on Conceptual Captions dataset [51]. Transformer captions are consistently more accurate (first and last images) and expressive (third image), while the model is able to capture the context better (second image). Figure from [51].

2.2 Reinforcement learning for sequence training

Previously discussed methods are commonly trained using back-propagation [6] in order to maximise the likelihood of the next word in the sentence given the previous ones. This has been discussed to produce exposure bias [47], which results in error accumulation at test time.

Recently, Reinforcement Learning (RL) [54] techniques have been shown to address the issue. Concretely, Ranzato et al. [47] have applied the REINFORCE algorithm [63] to directly optimise the non-differentiable test metrics captioning models use, taking care of the exposure bias. Still, this method is typically unstable during training if no context-dependent normalisation is performed. This behaviour is attributed to the high variance of the expected gradient while using mini-batch training.

Several methods have been proposed in order to provide a solution. One of them is Actor-critic [54], which trains a second network (the critic) to provide an estimate of the value of each generated word given the policy of the model (the actor). We can also find applications of Actor-critic for sequence problems [4].

Another simpler method also applicable to sequence generation is Self-critical Sequence Training (SCST) [48]. Starting from the REINFORCE algorithm, contrary to estimating the reward or its normalisation, SCST uses its own test-time inference output to normalise the reward signal. This results in a lighter model, as it does not require additional networks to be trained, and more robust training dynamics, as the gradient variance while using mini-batches is greatly reduced. Figure 2.10 shows a picture and its respective captions for a model trained with cross-entropy and the same with SCST. It can be appreciated that SCST training returns a more accurate and more detailed summary of the image.

2.3 Commonly used captioning datasets

In order to train machine learning algorithms, data is needed. Data is the power that fuels the optimisation and refinement of each and every one of these algorithms, and the more data and the better its quality, the better the training will be. Sometimes for small or very new research fields, public data is not available and it takes the carrier of the research to produce new datasets.

Luckily for computer vision, there are plenty of high-quality datasets

1. a blue of a building with a blue umbrella on it -1.234499
2. a blue of a building with a blue and blue umbrella -1.253700
3. a blue of a building with a blue umbrella -1.261105
4. a blue of a building with a blue and a blue umbrella on top of it -1.277339
5. a blue of a building with a blue and a blue umbrella -1.280045

(a) Ensemble of 4 Attention models (Att2in) trained with XE.

1. a blue boat is sitting on the side of a building -0.194627
2. a blue street sign on the side of a building -0.224760
3. a blue umbrella sitting on top of a building -0.243250
4. a blue boat sitting on the side of a building -0.248849
5. a blue boat is sitting on the side of a city street -0.265613

(b) Ensemble of 4 Attention models (Att2in) trained with SCST.

Figure 2.10: Top five captions generated by an ensemble of Att2in attention models trained with SCST as described in [48] and the MS COCO dataset (Section 2.3.1). The average log probability of the words in the caption is reported beside each caption. Figure from [48].

Figure 2.11: Example of (a) iconic object images, (b) iconic scene images, and (c) non-iconic images.

publicly available. Some of them are simply a collection of samples related to a problem, but some others spot a flaw in current research and have been crafted specifically to target it. Both types help the future research and promote investigation on solutions to those targeted problems. In this section, several image and video captioning datasets are presented, explaining their objective and showing samples of them.

2.3.1 MS COCO

The first dataset is Microsoft Common Objects in Context (MS COCO or COCO) [11, 37]. It is aimed to push the state of the art forward in object recognition, by providing context to the image via scene understanding. The authors argue [37] that non-ideal views of objects are more common in the real world, such as partially hidden or cluttered objects in a scene. Non-ideal (or iconic) examples can be found in Figure 2.11. Therefore contextual reasoning and spatial understanding is a key component for high-performance object recognition systems.

Accordingly, MS COCO contains 328000 instances of non-ideal views and context-related object images. These objects are grouped in 91 categories, with 82 of these having more than 5000 labelled instances. The instances are represented in the image as a segmentation mask, shown in Figure 2.12. This finally adds up to 2500000 instances in the dataset.

Regarding captions, there is a total of one million human-generated captions, gathered by crowd-sourcing using Amazon Mechanical Turk [9]. Each image has five captions and the images have been divided into 82783 samples for the training set, 40504 for validation and 40775 for the test set. A sample of the annotations can be found in Figure 2.13. The crowd-sourced annotators were provided a number of requisites for the descriptions:

- Describe all the important parts of the scene.

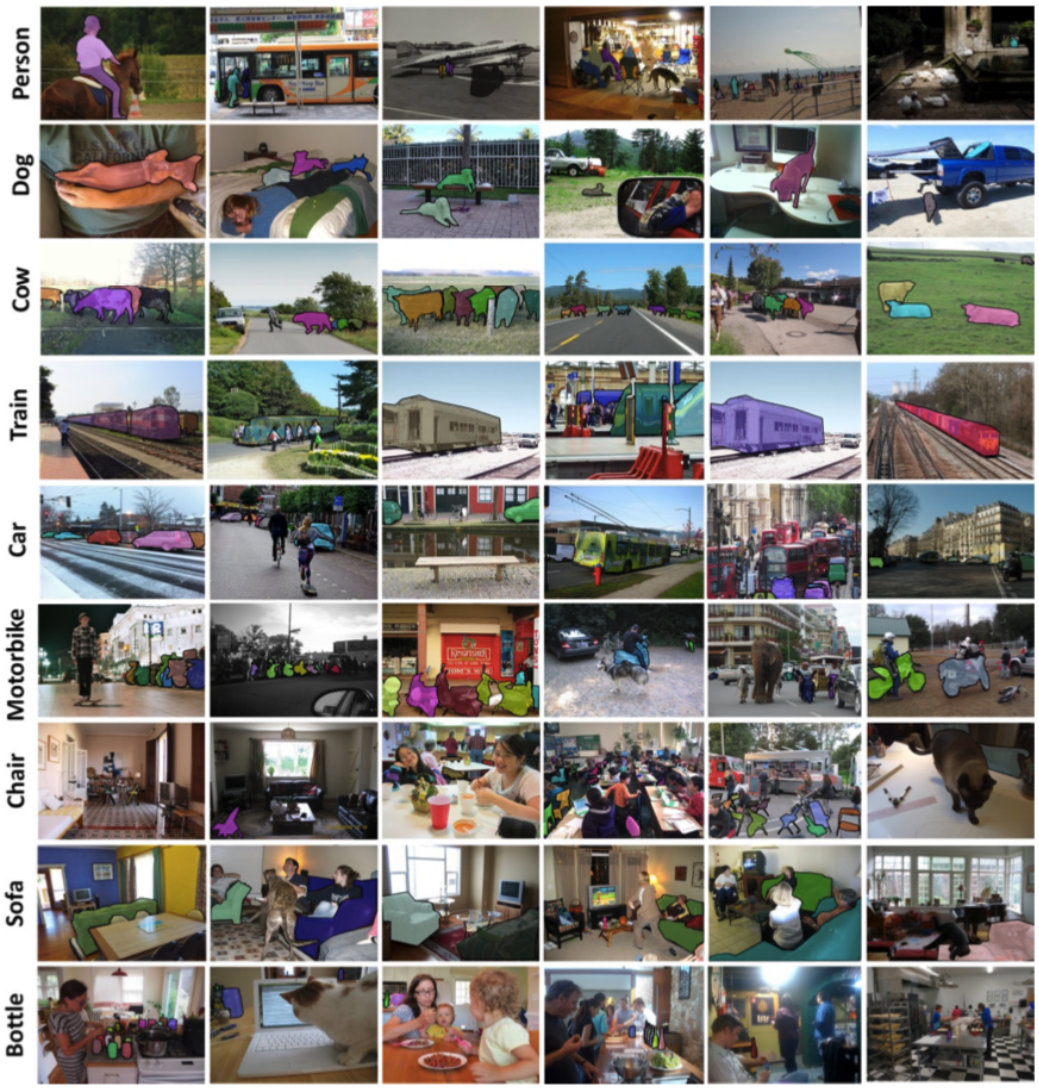


Figure 2.12: Samples of annotated images with segment masks in the MS COCO dataset.

Figure 2.13: Example images and captions from the Microsoft COCO Caption dataset.

- Do not start the sentences with "There is".
- Do not describe unimportant details.
- Do not describe things that might have happened in the future or past.
- Do not describe what a person might say.
- Do not give people proper names.
- The sentences should be at least eight words long.

2.3.2 TGIF

Tumblr GIF (TGIF) [36] is a dataset of animated GIFs or video clips. Its authors wanted to shed light on the lack of methods to automatically describe image sequences.

The dataset consists of 100000 animated GIFs from Tumblr and 120000 natural language descriptions created by crowd-sourcing. Cartoon, static or text-covered (memes) GIFs have been filtered out and each sample has one description. A sample of these can be found in Figure 2.14. Concretely, the training split has 80000 samples with one description, the validation split has 10708 samples with one description, and the test split 11360 with three descriptions each.

2.3.3 TRECVID VTT

The TREC Video Retrieval Evaluation (TRECVID) [2] is an annual workshop sponsored by the American National Institute of Standards and Technology (NIST). The goal of this workshop is to encourage research in information retrieval by providing large data collections, uniform scoring procedures, and a forum to compare results. Several tasks are offered, such as Video Search, Surveillance event detection, Instance search, etc in the interest of research in automatic segmentation, indexing, content-based retrieval, and more.

In particular, Video to Text (VTT) description task was introduced in 2016 to address matching and describing videos using textual descriptions. Each year, a subset of 2000 Twitter Vine videos is released and annotated five or less times by different annotators. So far, there have been three editions; 2016, 2017 and 2018, so a total of around 6000 videos have been released. Figure 2.15 shows some video and caption samples.

Annotators were asked, if possible, to include and combine in one sentence:

- **Who** is the video describing, such as concrete objects and beings (kinds of persons, animals, things)
- **What** are the objects and beings doing? (generic actions, conditions/state or events)
- **Where**, such as locale, site, place, geographic, architectural (kind of place, geographic or architectural)
- **When**, such as time of day, season, ...

2.3.4 Dataset summary

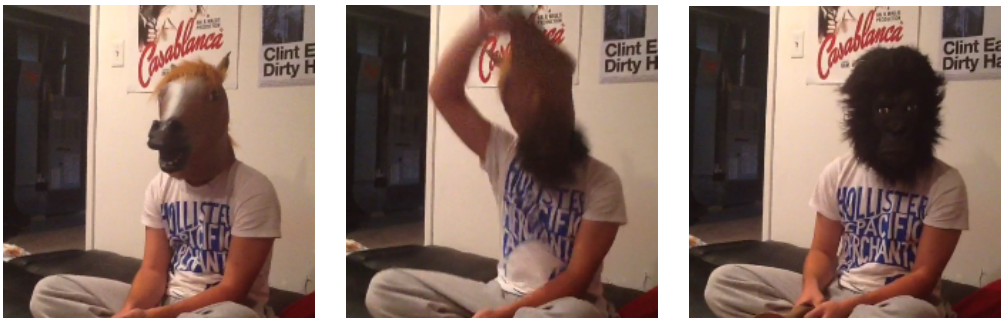
A small summary of all the presented datasets is shown in Table 2.1.

a woman in a car is singing.

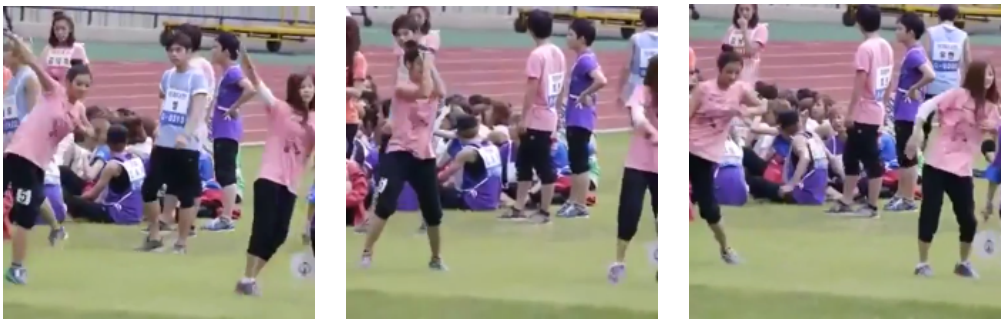
a man with a microphone is drinking a beverage.

a person is waving and smiling while looking to the right.

Figure 2.14: Example GIFs and captions from the TGIF dataset.



In a college dormitory room with posters for Casablanca and Dirty Harry on the white wall, a student on a black-covered bed sits cross-legged with a rubber head of a horse on until he rips it off to show the hairy black head of a gorilla beneath.



Cheerleaders practice before game starts.

An Irish setter wears a navy hoodie which covers his eyes as he walks across a suburban driveway and onto the grass.

Figure 2.15: Example videos and captions from the TRECVID VTT 2018 dataset.

Dataset	Samples	descr/sample
MS COCO	328000	5
TGIF	102068	1.23
TRECVID 2016	5880	2
TRECVID 2017	1880	3.5
TRECVID 2018	2000	3.5

Table 2.1: Dataset statistics.

Chapter 3

Methods

This chapter describes the essential parts and procedures for visual captioning. First, the used model architecture is detailed. Next, it is explained how the model learns, by using the two different loss functions. Then, the input of the network is described, by specifying how the raw data is processed. Finally, an explanation of the language metrics used to score the final results is provided.

3.1 Captioning model

Image or video captioning involves generating a human-readable textual description (a caption) given a picture, or a sequence of pictures. It is an easy task for a human, but a challenging one for a machine. As it involves understanding the content of an image, i.e. its context or the actions being represented, and learning to translate this understanding into natural language.

3.1.1 Description generation

Neural network solutions involve two main elements:

(1) *Feature extractor*, a model that is able to extract salient features from the input. These features are a latent representation of the image or video, so their richness will directly impact the overall performance of the model. Deep convolutional neural networks are common as feature extractors, particularly the use of top-performing models on image recognition tasks as pretrained extractors is a popular option. Our extractors of choice are described in Section 3.4.

(2) A *Language model*, which predicts the probability of the next word

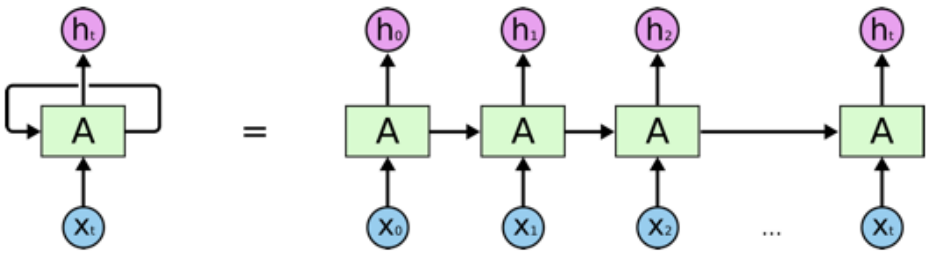


Figure 3.1: A recurrent neural network can be thought of as multiple copies of the same network, each passing a message to a successor. Image from [42].

to be generated in the description given the words previously generated. For this application, the model is another network able to produce the description given the extracted image or video features. In our case, a Long Short-Term Memory network (LSTM) [19] (a Recurrent Neural Network (RNN) variant) was selected.

A generic RNN architecture is depicted in Figure 3.1. Specifically, LSTM has a mechanism of sigmoid gates to control the memory cell $\mathbf{c}_t$, shown in Figure 3.2. At a timestep t, the network receives an input $\mathbf{x}_t$, the previous hidden state $\mathbf{h}_{t-1}$ and the previous memory cell state $\mathbf{c}_{t-1}$ and updates its values as follows:

$$\begin{aligned}
\mathbf{i}_t &= \sigma\left(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{b}_i\right) \\
\mathbf{f}_t &= \sigma\left(\mathbf{W}_{xf}\mathbf{x}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1} + \mathbf{b}_f\right) \\
\mathbf{o}_t &= \sigma\left(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{b}_o\right) \\
\mathbf{g}_t &= \tanh\left(\mathbf{W}_{xc}\mathbf{x}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1} + \mathbf{b}_c\right) \\
\mathbf{c}_t &= \mathbf{f}_t * \mathbf{c}_{t-1} + \mathbf{i}_t * \mathbf{g}_t \\
\mathbf{h}_t &= \mathbf{o}_t * \tanh\left(\mathbf{c}_t\right),
\end{aligned} \tag{3.1}$$

where $\mathbf{W}$ and $\mathbf{b}$ are the weights and biases, σ is the sigmoid function and $*$ is the element-wise product of two vectors. The next word is predicted using the Softmax function as

$$\mathcal{F}\left(\mathbf{p}_{ti} \mid \mathbf{W}_s, \mathbf{b}_s\right) = \frac{\exp\left(\mathbf{W}_s\mathbf{h}_{ti} + \mathbf{b}_s\right)}{\sum_{j=1}^{K}\exp\left(\mathbf{W}_s\mathbf{h}_{tj} + \mathbf{b}_s\right)}, \tag{3.2}$$

where $\mathbf{p}_{ti}$ is the probability distribution over the vocabulary for the next predicted word i.

Finally, the ground-truth data, in this case sentence descriptions, undergoes a preprocessing step. First, each word is assigned an index and replaced with it (word2index [66]). Then, each word coming either from the ground

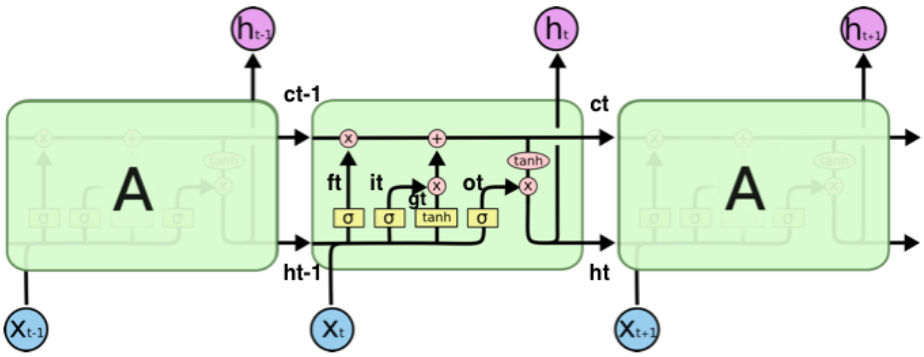


Figure 3.2: LSTM architecture, unrolled. It forms a chain just like any RNN, but has four specific network layers, the forget gate $\mathbf{f}_t$, input gate $\mathbf{i}_t$, cell state $\mathbf{c}_t$ and the output gate $\mathbf{o}_t$. Image from [42].

truth or generated from a previous generation step is further encoded semantically by using a word embedding [56], and fed to the RNN to produce the following word. This process ends when the networks outputs and end-of-sentence (EOS) token or the maximum length of the sequence is reached.

3.1.2 Encoder-decoder

The encoder-decoder architecture [60] is a popular way to structure the subnetworks aforementioned, where each component is trained jointly. It is also possible to pretrain one, use already precomputed features of the dataset, etc. In our case, dataset features are computed beforehand and loaded into memory during training.

The beauty of this approach is that a single end-to-end model can be trained on the problem, taking the raw data and outputting the result, with no intermediate steps. Figure 3.3 depicts the final architecture.

The probability of the correct description can then be modelled using the chain rule over $y_1, \ldots, y_T$, where T is the length of the sentence

$$\log p_\theta(Y \mid I) = \sum_{t=1}^{T} \log p_\theta \left(y_t \mid I, y_1, \ldots, y_T\right). \tag{3.3}$$

At training time, (Y, I) is a pair (sentence description, image). It is possible to model $p_\theta \left(y_t \mid I, y_1, \ldots, y_T\right)$ using an RNN so, as mentioned before, LSTM is picked. The model equations during training therefore read

$$\begin{aligned} \mathbf{x}_0 &= \mathrm{CNN}(I) \\ \mathbf{x}_t &= W_e y_t, \quad t \in \{1, \ldots, T\} \\ p_{t+1} &= \mathrm{LSTM}\left(\mathbf{x}_t\right), \quad t \in \{1, \ldots, T\} \end{aligned} \tag{3.4}$$

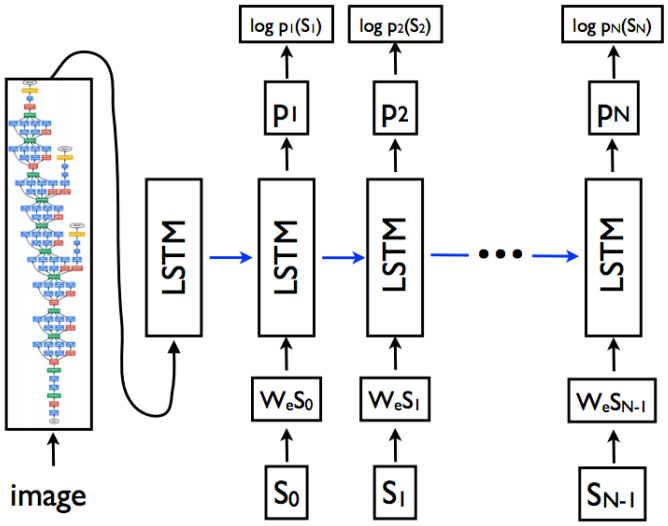


Figure 3.3: LSTM model combined with a CNN image embedder (as defined in (3.4)) and word embeddings. The unrolled connections between the LSTM memories are in blue and they correspond to the recurrent connections. Image from [60].

Let CNN be the image embedder, W_e the word embedder. The decoder takes the last state from the encoder, $\mathbf{x}_0$ to start generating the output $\hat{Y} = \{\arg\max p_1, \ldots, \arg\max p_T\}$, based on the previous word $\mathbf{x}_t$ and the ground truth y_t. Additionally, the decoder can take a vector $\mathbf{h}_t$ representing the hidden states of the RNN.

3.2 Cross-entropy loss

In order to train a sequence model so that its output distribution approximates the target distribution at each decoding step t, we can apply the generally used teacher forcing algorithm [6]. This algorithm maximises the log-likelihood of the model output X to match the ground truth $y = \{y_1, y_2, \cdots, y_T\}$, by minimising the cross-entropy objective

$$\mathcal{L}_{CE} = -\sum_{t=1}^{T} \log p_\theta \left(y_t \mid y_{t-1}, \mathbf{h}_{t-1}, X\right). \tag{3.5}$$

Let $\mathbf{h}_{t-1}$ be the hidden state of the RNN from the previous step and p_θ the probability of an output parametrised by θ. In inference time, the output

can be produced simply by greedy sampling of the sequence being generated

$$\hat{y}_t = \arg\max_y p_\theta \left(y \mid \hat{y}_{t-1}, \mathbf{h}_{t-1} \right), \tag{3.6}$$

with $\hat{y}_t$ as the token generated by the model at time t.

3.3 Reinforcement learning loss

Nonetheless, cross-entropy loss does not always produce the best results for the evaluation metrics. The first issue is *exposure bias*, produced by the decoder using two inputs, the previous state $\mathbf{s}_{t-1}$ and the ground truth y_t, to calculate the current state $\mathbf{s}_t$ and thus the next token $\hat{y}_t$. The problem comes at test time when the input for the next state comes only from the distribution of the previous state, and not from the ground truth one. This produces a cumulative error while generating the output.

Secondly, the model is trained using cross-entropy loss, although is evaluated in test time using (non-differentiable) natural language processing metrics such as BLEU or CIDEr (Section 3.5). This creates a *mismatch* between train and test measurements that can eventually produce inconsistent results.

Recently, a problem reformulation using policy gradient algorithms [26] has resulted in effectively addressing these issues. It no longer uses the ground truth in the decoding stage and enables NLP metrics to have a direct impact on the training via the reward function.

3.3.1 REINFORCE

In Reinforcement Learning, an agent takes actions according to a policy π. Depending on the application, the policy can be modelled differently. In our case, a captioning system can be represented by a language model $p(y \mid X)$ where y is the output produced given the input X, represented by e.g. an RNN following a policy π_θ. The output of this RNN is passed through a softmax function at each step, producing the output $\hat{y}_t$. This sequence of actions (here also regarded as sentence decoding) continues until the maximum length of the sequence is reached or an end-of-sequence (EOS) token is produced.

Once the end is reached, the produced sequence $\hat{y}_t$ is compared against the ground truth y_t using a reward function. It is worth noting that the reward for the whole sequence of actions of the agent is only given when the sequence is finished. This can be argued to be a constraint, as evaluating subsets or unfinished sentences would not make sense.

The goal is to maximise the reward of the model by finding optimal policy parameters θ. We can define an optimisation loss function as the negative expected reward of the full sequence,

$$\mathcal{L}_\theta = -\mathbb{E}_{\hat{y}_1,\cdots,\hat{y}_T\sim\pi_\theta(\hat{y}_1,\cdots,\hat{y}_T)}\left[r\left(\hat{y}_1,\cdots,\hat{y}_T\right)\right], \tag{3.7}$$

where $\hat{y}_t$ are the actions from 1 to the maximum T, and r is the reward function. In practice, the gradient is usually approximated by producing only one caption. Then, the gradient of the loss can be calculated by taking derivatives,

$$\nabla_\theta \mathcal{L}_\theta = -\mathop{\mathbb{E}}_{\hat{y}_{1\cdots T}\sim\pi_\theta}\left[\nabla_\theta \log \pi_\theta\left(\hat{y}_{1\cdots T}\right) r\left(\hat{y}_{1\cdots T}\right)\right]. \tag{3.8}$$

Using the chain rule, this equation can be rewritten [67] as

$$\nabla_\theta \mathcal{L}_\theta = \frac{\partial \mathcal{L}_\theta}{\partial \theta} = \sum_t \frac{\partial \mathcal{L}_\theta}{\partial \mathbf{o}_t}\frac{\partial \mathbf{o}_t}{\partial \theta}, \tag{3.9}$$

with $\mathbf{o}_t$ as the output of the RNN, right before passing it to the softmax function. The gradient is thus expressed as [63, 67]:

$$\frac{\partial \mathcal{L}_\theta}{\partial \mathbf{o}_t} = \left(\pi_\theta\left(y_t \mid \hat{y}_{t-1}, \mathbf{s}_t, \mathbf{h}_{t-1}\right) - \mathbf{1}\left(\hat{y}_t\right)\right)\left(r\left(\hat{y}_1,\cdots,\hat{y}_T\right) - r_b\right), \tag{3.10}$$

where $\mathbf{h}_t$ is the hidden state of the RNN, $\mathbf{1}(\cdot)$ represents a one-hot encoding with the dimensionality being the number of words in the vocabulary, and r_b is a baseline reward that can take any value that does not depend on θ.

The purpose of r_b is to push the model to select actions that produce $r > r_b$ and discourage the opposite. Having this baseline also induces a more stable training, by reducing the variance of the gradient estimator [63]. We can show that this baseline will not affect the gradient by

$$\begin{aligned}\mathbb{E}_{\hat{y}_{1\cdots T}\sim\pi_\theta}\left[\nabla_\theta \log \pi_\theta\left(\hat{y}_{1\cdots T}\right) r_b\right] &= r_b \sum_{\hat{y}_{1\cdots T}} \nabla_\theta \pi_\theta\left(\hat{y}_{1\cdots T}\right) \\ &= r_b \nabla_\theta \sum_{\hat{y}_{1\cdots T}} \pi_\theta\left(\hat{y}_{1\cdots T}\right) \\ &= r_b \nabla_\theta 1 = 0.\end{aligned} \tag{3.11}$$

This algorithm is called *REINFORCE* [63], and it is a policy gradient solution for sequence-to-sequence problems.

3.3.2 Self-critic

Nevertheless, REINFORCE is still suffering from high variance due to using only one sample at each training step. One proposed solution to this is the Self-Critic algorithm [48], where instead of estimating the baseline using current samples, the output at inference time of the model is used, normally applying greedy search. The loss is then modified as

$$\begin{aligned}\mathcal{L}_\theta = \frac{1}{N}\sum_{i=1}^{N}\sum_t \log \pi_\theta\left(\hat{y}_{i,t} \mid \hat{y}_{i,t-1}, \mathbf{s}_{i,t}, \mathbf{h}_{i,t-1}\right) \\ \cdot \left(r\left(\hat{y}_{i,1}, \cdots, \hat{y}_{i,T}\right) - r\left(\hat{y}^g_{i,1}, \cdots, \hat{y}^g_{i,T}\right)\right)\end{aligned}, \tag{3.12}$$

where $\hat{y}^g_{i,t}$ is the greedy selection of the final output distribution for timestep t.

The aforementioned problem of observing the reward only after the whole sequence of actions is sampled is still present. If the model chooses a bad action, it will not realise it until the end. This is especially acute when the action is taken randomly at the beginning of the training. In order to palliate it, it has been suggested [47] to use pretrained models with cross-entropy loss as initialisation.

3.4 Feature extraction

In machine learning, the feature extraction from input data is rarely done from scratch anymore, and instead, previously-trained extractors are applied using transfer learning. The idea of transfer learning [55] is to use knowledge acquired from one task to solve related ones. It also significantly reduces the need for training data and time in the target domain that it is applied to.

Specifically, we apply network-based deep transfer learning [55] by partially reusing the network pretrained in a different domain, including the architecture and the parameters. The assumption is that a network, mimicking the human brain, produces iterative levels of abstraction. Consequently, after the training, one could export the learned concepts to solve different problems.

In this thesis, successful classification models are used for the purpose of extracting high-level features necessary for input comprehension.

3.4.1 Residual Network (ResNet)

The universal approximation theorem [20] states that a network with a single layer is enough to represent any function. However, in practice, this approach is prone to training problems like overfitting, so growing the depth

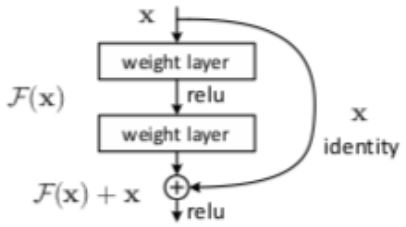


Figure 3.4: Residual learning: a building block. Image from [18].

of the network is a more popular approach. Deep neural networks are problematic alike, suffering from gradient vanishing. The authors of the Residual Networks (ResNets) [18] propose a shortcut connection $\mathcal{H}(\mathbf{x}) = \mathcal{F}(\mathbf{x}) - \mathbf{x}$ named *residual connection*, pictured in Figure 3.4. This allows easier flow of information during training, thus enabling a greatly larger number of layers in the network and consequently achieving much better performance.

The authors applied this method to image recognition, achieving the first place in the ILSVRC 2015 image classification competition and a number of other related ones. To this day, ResNets have seen further studies and improvements, and keep being a popular option for Image Recognition tasks.

3.4.2 Inflated 3D Convolutional Network (I3D)

Inflated 3D Convolutional Network (I3D) [10] produces accurate video action classification using an original method. It makes use of notably performing image recognition models (2D) and inflate filters and kernels to 3D, thus creating an additional temporal dimension. The authors also discuss that their use of two-stream [15] configuration, using not only appearance (RGB) but also motion information (Optical Flow), considerably benefits the overall performance. Figure 3.5 depicts a diagram of the network. Concretely, the base network used is ImageNet-pretrained Inception-V1 [22].

3.5 NLP metrics

Normally, machine learning and even natural language processing tasks are easy to evaluate since a large number of them require simple label matching. The performance on these tasks can be simply assessed using precision, recall, F-score, and even accuracy.

However, natural language generation results are much more complex to evaluate. It consists of a set of *candidate* texts and a set of *references*. If a candidate text A is closer to a reference text than candidate B, A will have a

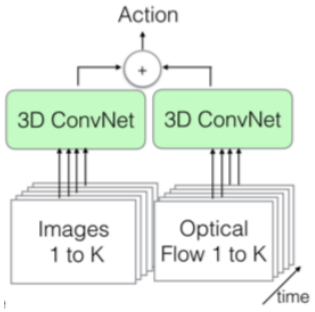


Figure 3.5: Two-Steam 3D Convolutional Network diagram of I3D. K is the total number of frames in a video. Image from [10].

higher score. This closeness computation typically uses precision and recall in order to match each candidate and reference pair.

To design a metric that is not computationally expensive, fast enough to use in real-life scenarios and that has significant human evaluation correlation is a difficult challenge. A number of metrics have been proposed, three of which are used in this thesis and will be explained next.

3.5.1 BLEU

The Bilingual Evaluation Understudy, BLEU [43], is one of the first and most popular metrics for sequence evaluation. It was designed for machine translation (MT) tasks, and it is fast and language-independent. BLEU accounts for precision and recall using modified n-gram precisions.

An n-gram is a sequence of n words, i.e. a 1-gram (unigram) can be "`potato`", and a bigram "`a potato`". n-gram precision is the ratio of n-grams in the candidate text which appears in *any* of the references. It is called "modified" because it only counts as many times as the number of occurrences in the references, therefore the count is *clipped*. It is formulated as

$$p_n = \frac{\sum_{C \in \{\text{Candidates}\}} \sum_{n\text{-gram} \in C} Count_{clip}(n\text{-gram})}{\sum_{C' \in \{\text{Candidates}\}} \sum_{n\text{-gram}' \in C'} Count\,(n\text{-gram}')}. \tag{3.13}$$

To include all the precisions, their geometric mean is taken and then multiplied with a *brevity penalty* factor BP

$$\text{BLEU-N} = \text{BP} \cdot \left(\prod_{n=1}^{N} p_n \right)^{\frac{1}{N}} \tag{3.14}$$

$$= \text{BP} \cdot \exp\left(\frac{1}{N} \sum_{n=1}^{N} \log p_n \right). \tag{3.15}$$

N-grams are used up to length N, being $N = 4$ normally. Which is indicated in the name of the metric in the form of BLEU-4.

The last part to define is the brevity penalty, this penalty compares the length of the candidate to the closest in length of all the references. In doing so, a candidate is penalised by this term if it is too short, and penalised by the modified precision if it is too long. The formulation is as follows

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases}, \tag{3.16}$$

with c being the total length of the candidate corpus and r the average length of all references.

3.5.2 METEOR

The next metric is Metric for Evaluation of Translation with Explicit Ordering (METEOR) [34]. This metric reports better human correlation than BLEU, as the brevity penalty BLEU uses length averaged over the entire corpus, which provokes inaccurate individual scores.

METEOR solves this problem by first replacing the precision computation with a weighted F-score (harmonic mean). A map is created that can be aligned between the candidate and reference texts, first by exact matching, then using Porter stemmer [46], and finally using WordNet [39] synonyms. Let the precision P be the number of unigrams mapped divided by the reference length (number of reference unigrams), and the recall R be the number of unigrams mapped divided by the candidate length (number of candidate unigrams). The F-score is then

$$\text{Fmean} = \frac{10PR}{R + 9P}. \tag{3.17}$$

Second, the penalty function takes the word order into consideration. Its computation is based on chunks, and a chunk is the longest possible matched n-gram. Thus, the longer the n-grams, the lower the number of chunks,

$$\text{Penalty } = 0.5 \cdot \left(\frac{\#\text{ chunks}}{\#\text{ unigrams\_matched}} \right)^3. \tag{3.18}$$

The final METEOR score is computed as

$$\text{METEOR} = \text{Fmean} \cdot (1 - \text{Penalty}). \tag{3.19}$$

3.5.3 CIDEr and CIDEr-D

In Consensus-based Image Description Evaluation (CIDEr) [58], the authors claim that regarding human judgement, what humans like occasionally does not match with what is human-like. Therefore, they introduce a consensus-based metric in which a candidate is measured as to how the majority of the people produced the references.

In order to achieve this, all reference and candidate sentences are stemmed and transformed into a set of n-grams. Next, to measure how often n-grams appear as well as noting that very common n-grams should have lower importance, a Term Frequency Inverse Document Frequency (TF-IDF) [49] weighting for each n-gram is computed.

Lastly, the average cosine similarity is used between the candidate and reference sentences for n-grams of length n, as follows

$$\text{CIDEr}_n\left(c_i, S_i\right) = \frac{1}{m} \sum_{j=1}^{m} \frac{\boldsymbol{g}^n\left(c_i\right) \cdot \boldsymbol{g}^n\left(s_{ij}\right)}{\left\|\boldsymbol{g}^n\left(c_i\right)\right\| \left\|\boldsymbol{g}^n\left(s_{ij}\right)\right\|}, \tag{3.20}$$

where m is the number of reference sentences, $\boldsymbol{g}^n(\cdot)$ is a vector formed by $g_k(\cdot)$ (the TF-IDF weighting) corresponding to all n-grams of length n, $\|\cdot\|$ is the magnitude of a vector, c_i and s_{ij} correspond to the candidate and reference sentences, respectively.

The scores from several n-grams are combined as

$$\text{CIDEr}\left(c_i, S_i\right) = \sum_{n=1}^{N} w_n \, \text{CIDEr}_n\left(c_i, S_i\right), \tag{3.21}$$

with $w_n = 1/N$ again as uniform weight, and $N = 4$ which means that 1 to 4-grams are used.

A second version of CIDEr, called CIDEr-D, is also designed, which aims to remedy CIDEr score exploitation under gaming circumstances that would be otherwise poorly judged by humans. First, the stemming is removed in order to use the correct word form. Second, high-confidence words can be repeated to obtain a better score, for which a Gaussian penalty on the

difference between candidate and reference lengths is introduced. Third, as the length penalty can be tricked by repeating high-confidence words until the reference length is met, the numerator n-gram count is clipped. The resulting formulation is as follows:

$$\text{CIDEr-D}_n\left(c_i, S_i\right) = \frac{10}{m}\sum_{j=1}^{m} e^{\frac{-\left(l(c_i)-l\left(s_{ij}\right)\right)^2}{2\sigma^2}} \frac{\min\left(\boldsymbol{g}^n\left(c_i\right), \boldsymbol{g}^n\left(s_{ij}\right)\right) \cdot \boldsymbol{g}^n\left(s_{ij}\right)}{\left\|\boldsymbol{g}^n\left(c_i\right)\right\| \left\|\boldsymbol{g}^n\left(s_{ij}\right)\right\|}, \tag{3.22}$$

where $l(\cdot)$ is the length of a sentence and $\sigma = 6$.

The final CIDEr-D metric is computed similarly to CIDEr (3.21) as

$$\text{CIDEr-D}\left(c_i, S_i\right) = \sum_{n=1}^{N} w_n \,\text{CIDEr-D}_n\left(c_i, S_i\right). \tag{3.23}$$

Chapter 4

Experimental setup

This section discusses the configuration details of each component of the architecture so that the experiments could be successfully replicated. First, the datasets selected and their combination are discussed. Next, specifics about the features extracted from those datasets are shared, followed by the vocabulary used by the language model to generate the output. Lastly, architectural and implementation details are presented.

4.1 Datasets

The datasets were selected based on the comparison of a metric performance previously conducted in the research group [52]. A number of different annotated image and video datasets were used, both alone and in combination, resulting in the best performing combination of MS COCO (Section 2.3.1) and TGIF (Section 2.3.2).

The training dataset consisted of a concatenation of those two, with 208496 samples and 539826 human-annotated descriptions. For validation and test, datasets from TRECVID (Section 2.3.3) were used, as the system described in this thesis is aimed to compete in their VTT challenge in 2019. Concretely for validation, the test split from the 2017 edition was used, with 1880 samples and 5274 descriptions. The test dataset was formed by 2018 edition test split data, with 1904 instances and 9520 captions.

4.2 Features

The features extracted from the datasets were multimodal. Previous studies implemented of the research group [52] suggested better performance by mixing different features from the same data, in addition to having different

perspectives using more modalities available than only standalone frames or video. In this thesis, image features are adopted by making use of ResNet feature extractor, explained in Section 3.4.1, as well as video features extracted by I3D, explained in Section 3.4.2.

For image feature extraction, two sizes of the network were used, namely ResNet-101 and ResNet-152. In order to summarise a video input into a picture, the heuristic consisted in taking the middle frame. A 2048-dimensional feature vector is extracted from the 5th average pool layer. This layer is the last one prior to the fully-connected output, thus having the features with the highest abstraction level.

Regarding video features, the videos were first resampled to 25 frames per second as in the original I3D paper and 128 frames were taken from the center. The extractor is applied convolutionally over the whole video and the output is average-pooled in order to produce a 2048-dimensional feature vector.

From these extractions, two sets of features are assembled. First, an image-only feature set is constituted by the concatenation of the output of ResNet-101 and ResNet-152, producing a vector of dimension 4096. This set will be also referred to as "image-only features". Second, an image-video feature set is formed by the concatenated output of ResNet-152 and I3D, also having a final dimension of 4096. This set will be referred to as "image+video features".

4.3 Vocabulary

The vocabulary was based on the words from the annotated descriptions provided in the datasets, in order to learn to imitate the same expressions as humans generate. A performance comparison was previously done in the research group [52], reaching the conclusion that using MS COCO and TGIF datasets, the best vocabulary was produced by constraining it to words appearing only in these two.

The decoder uses a vocabulary that consists only on words that appeared in MS COCO ground truth captions. Only words with a threshold frequency of greater or equal to 4 were taken, generating a vocabulary of 9956 tokens. Previous experiments performed with a bigger, joint vocabulary of MS COCO and TGIF with 11679 tokens resulted in worse performance. Therefore the vocabulary was not varied for the rest of the experiments.

4.4 Implementation

The models and the data pipeline were implemented using the PyTorch [45] framework. All parameters not specified here were using their respective default values. A public implementation can be found in the DeepCaption repository [17].

For the encoder-decoder architecture, the encoders are directly constituted by the feature extractors discussed in Section 4.2. Regarding the decoder, the features of batch size 128 pass through an embedding layer of dimensionality 512, followed by a 2-layer LSTM with 1024 hidden units. The dropout regularisation applied in the input and the LSTM is using a probability $p = 0.5$. Gradient clip of 0.1 was applied. The optimiser used during the cross-entropy training stage is centered RMSprop, with a learning rate of 0.001 and weight decay (L_2 penalty) of 10^{-6}, while for the self-critical stage Adam was used with a learning rate of 5×10^{-5}. Gradient clipping and weight decay were optimised on the hyperparameter search for this part of the training.

Chapter 5

Experiments and results

This chapter presents the experiments conducted in this thesis. Firstly, it is explained how hyperparameter search is carried out in order to find the best values for the specific model and data used. Then, the next section explains each experiment and provides training dynamics and individual results of the models outlined in Chapter 3 and configurations explained in Chapter 4. Following that, the performance of the generated descriptions is quantitatively compared via the selected NLP metrics and qualitatively assessed by a collective comparison against human descriptions. After that, a brief study is shown on how the automatic metrics can be abused to produce high scores with simple descriptions, as an illustration of bad use of the self-critical training. The chapter concludes with a reflection on the lack of proper stopping criteria for this reinforcement learning training method.

5.1 Hyperparameter search

A number of hyperparameter searches have been performed in order to optimise the architecture to find the best possible model trained under self-critic loss (Section 3.3.2). The current architecture and architectural parameters were mainly fixed in order to establish a reliable baseline on which to compare the different self-critic loss experiments. The parameter search was conducted sequentially, and findings were aggregated for the next parameter. All targeted parameters and their range or options are listed in Table 5.1.

Weight decay and gradient clipping were optimised due to the discrepancy between their current value and the paper statement of suppressing them from the self-critical training. The epoch when to switch from cross-entropy training to reinforcement learning is also optimised for this task. It is interesting to see if a switching moment other than the best one from the

Table 5.1: Hyperparameters to be optimised for self-critical training. Squared brackets on the range mean uniformly selected through the integer range, while curly braces denote only those options.

Hyperparameter	**Range**
Gradient clip	{0, 0.1}
Weight decay	$\{0, 10^{-6}\}$
Initial epoch	[5, 16]

validation loss or from the automatic metrics generates a better agent. The rationale for this search is that using slightly under or overtrained models may encourage exploration in early stages of the next training stage, rather than direct exploitation, as the paradigm shifts from cross-entropy to self-critical learning.

5.1.1 Results

Hyperparameter search results for self-critical training are displayed here. In the case of discrepancies among score improvement, CIDEr will be the deciding metric.

Gradient clipping is a technique used to mitigate numerical over or underflow when large gradient updates are performed [44]. This misbehaviour is the most notorious when recurrent neural networks are employed. Gradient clipping is also enabled in the original self-critic loss paper [48], so an empirical comparison is put in place, seen in Table 5.2. It concludes that the clipping is beneficial for the training.

Table 5.2: Gradient clip hyperparameter search.

clip	**CIDEr**	**CIDEr-D**	**METEOR**	**BLEU-4**
0	0.2856	0.0975	0.2065	**0.0285**
0.1	**0.2882**	**0.0995**	**0.2078**	0.0280

Weight decay as a regularisation technique [31] was applied in the baseline model, but empirical analysis showed that it was decreasing the model performance with the change of optimisation method. Scores are shown in Table 5.3.

Table 5.3: Weight decay regularisation hyperparameter search.

decay	CIDEr	CIDEr-D	METEOR	BLEU-4
10^{-6}	0.2882	**0.0995**	**0.2078**	0.0280
0	**0.2913**	0.0964	0.2059	**0.0296**

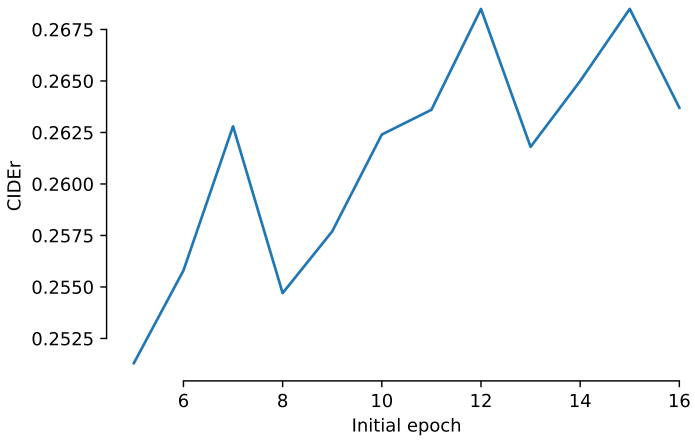


Figure 5.1: Initial epoch hyperparameter search.

Initial epoch selection is performed in order to assess the best epoch as the initialisation of the reinforcement learning training. It has been discussed [48] that this point is not necessarily the best scoring cross-entropy model, so a search was conducted for this concrete case and the results are displayed in Figure 5.1. It shows no particular point to be highlighted, therefore the selected epoch number 13 was a trade-off between the validation loss and the metric scores.

5.2 Experiment 1: Cross-entropy training

5.2.1 Setup

The first experiment constitutes a baseline for the next ones to compare with, therefore a conventional configuration is set up. The encoder-decoder architecture from Section 3.1.2 is used, fed with image-only features explained in Section 4.2. The model is optimised by minimising the widely-adopted cross-entropy loss function from Section 3.2. The hyperparameter optimisation was previously conducted in the research group [52] and those values were used, which is not the same as the hyperparameter search from Sec-

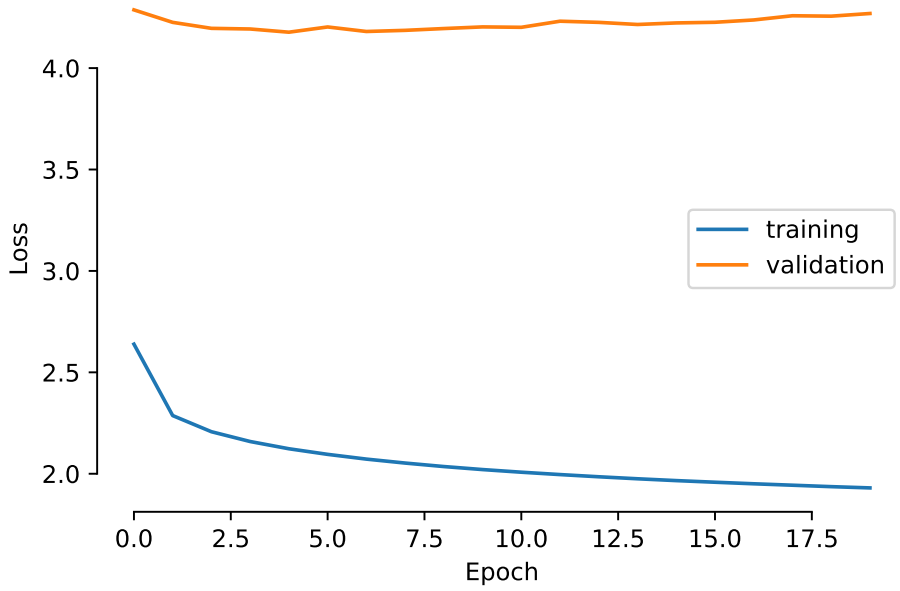


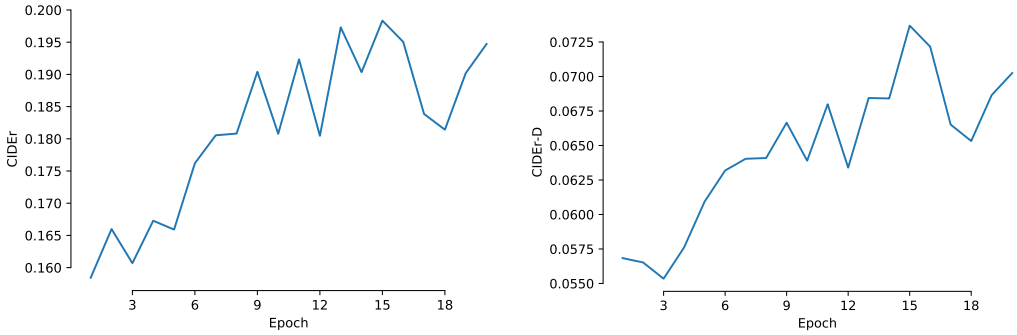
Figure 5.2: Learning curves for cross-entropy loss training with image features.

tion 5.1, performed for the self-critic loss in the following experiments. The parameters were therefore set as Adam optimiser [27] with learning rate 10^{-3} and weight decay of 10^{-6} and gradient clipping of 0.1.

5.2.2 Results

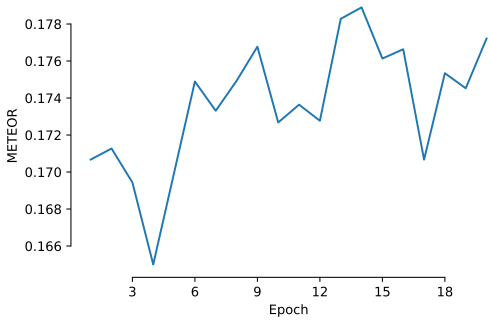
The progress of the training and validation loss of the model using cross-entropy is displayed in Figure 5.2. The score progression during the training can be also seen in Figure 5.3. Ordinary-looking learning curves can be observed.

Nevertheless, the model chosen as the pretrained initialisation for subsequent Reinforcement Learning models need not precisely be the one having the lowest loss on the validation dataset, as explained in Section 5.1. Some test data samples and the corresponding generated description are depicted in Figure 5.4, in which it is appreciated how the context is captured most of the times, but the action or the relations between the subjects are not highly accurate.

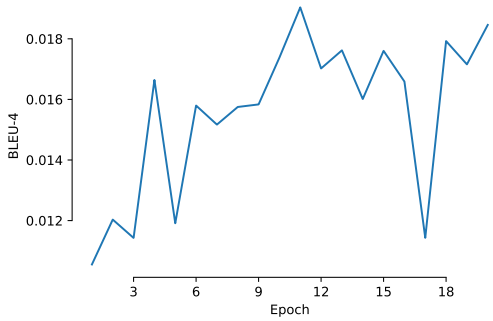


(a) CIDEr

(b) CIDEr-D

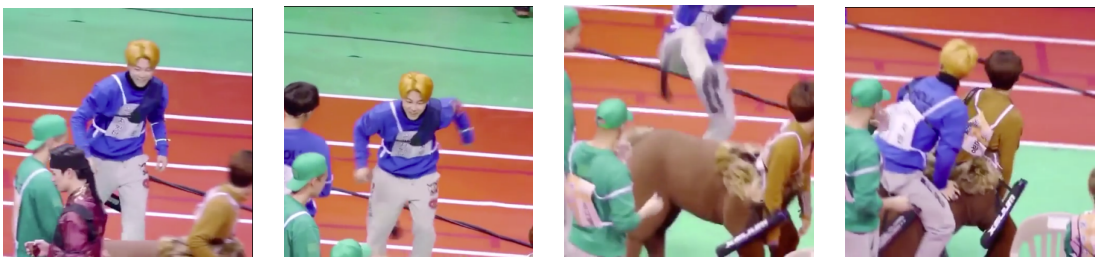


(c) METEOR



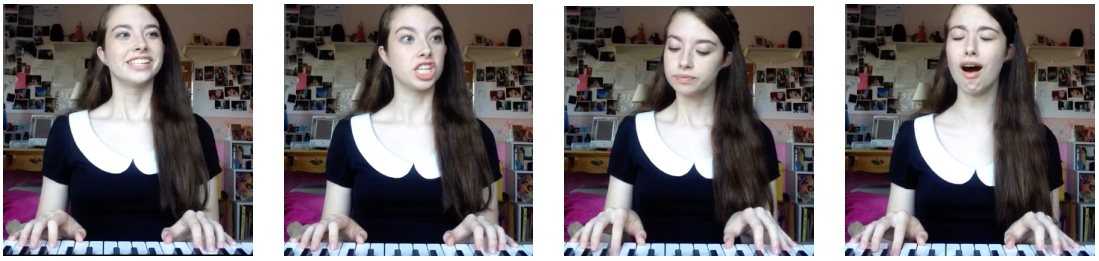
(d) BLEU-4

Figure 5.3: Metric scores on the test set for cross-entropy loss training with image-only features.



A wrestler is <unk> another wrestler down the mat

A man is singing and dancing with other men



A woman with long hair is smiling and talking

A man is jumping in the air and falls

Figure 5.4: Inference of video descriptions for the captioning model using cross-entropy loss and two sets of ResNet image features. Four frames were sampled from the video uniformly to facilitate understanding for the reader.

5.3 Experiment 2: Abusive self-critical training

5.3.1 Setup

This is the first experiment introducing the self-critic loss. Architecture, features and hyperparameters were inherited from the previous experiment. The only parameters changed were the ones under the study of the hyperparameter search from Section 5.1, namely the learning rate to $5 \cdot 10^{-5}$, no weight decay and no gradient clipping. The model is trained from epoch 13 (the best performing from the previous experiment), to epoch 50. The reward function used in the loss is CIDEr (Section 3.5.3). It is known to be a gameable metric and thus not producing qualitatively good results (Section 5.7), but the experiment is kept here for comparison and completeness purposes.

5.3.2 Results

The progress of the training and validation loss of the model using self-critic loss is displayed in Figure 5.5. The score progression during the training can be also seen in Figure 5.6. It can be observed that both figures look similar to a well performing model with a robust rewarding function, which is far from ideal because no flaws can be spotted during training.

Some data samples of the test dataset and the corresponding generated description are depicted in Figure 5.7. It shows how the context and the subject is still being captured correctly in the description, but the model produces garbage by repeating key n-grams in order to score better.

5.4 Experiment 3: Self-critical training

5.4.1 Setup

In this experiment, all configurations are inherited from the previous one. The only change is the reward function used for the loss, this being CIDEr-D, a derivation from CIDEr more robust to abuse/gaming. Due to its properties, the quality of the generated captions does not decrease exponentially, therefore allowing a longer training for as much as 120 epochs compared to the previous 50. A higher number would result in unmanageable training times for too small performance increments. The conducted hyperparameter search resulted in the same values as for the previous experiment.

A young girl with a girl is smiling and a girl with a woman is smiling and a girl is

A group of men are dancing in a group of people are dancing on a stage with a group of

A young boy holding a dog with a dog is holding a dog in a dog is a dog is

A young man wearing a red hat is dancing on a stage with a man in a man is dancing

Figure 5.7: Inference of video descriptions for the captioning model using ResNet image features and a training reward function that was not robust enough against score gaming. Four frames were sampled from the video uniformly to facilitate understanding for the reader.

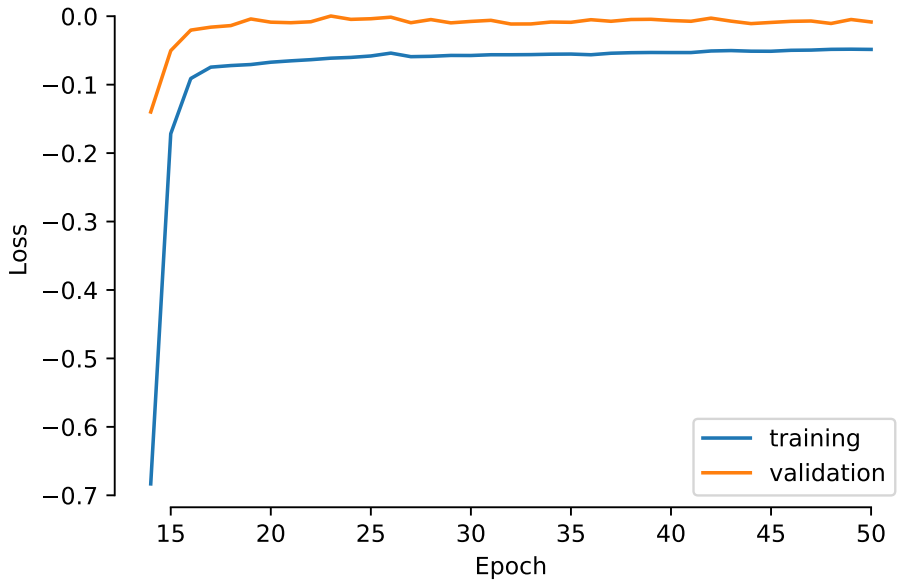
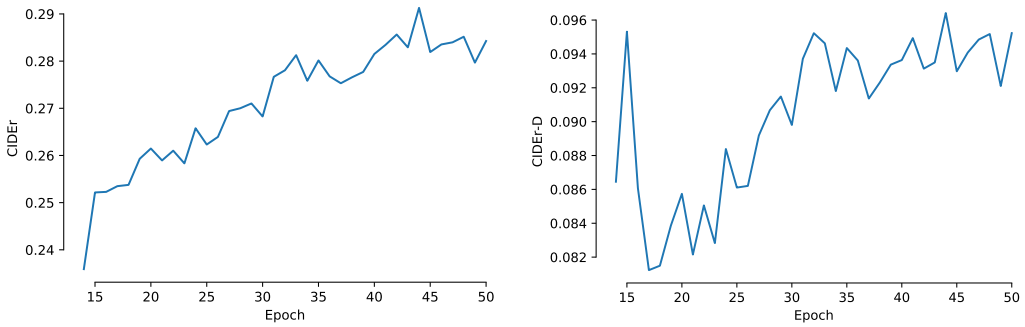


Figure 5.5: Learning curves for a training using self-critic loss with image-only features.

5.4.2 Results

The progress of the training and validation loss of the model using cross-entropy is displayed in Figure 5.8. The score progression during the training can be also seen in Figure 5.9. It can be observed that both losses tend asymptotically to zero, coming from the negative loss region. Additionally, scores on all metrics computed on the test set are moderately noisy, but never stop improving. This situation of uninformative metrics and learning curves presents a problem in several aspects, one of them is the lack of stopping criteria, which will be discussed in detail in Section 5.9.

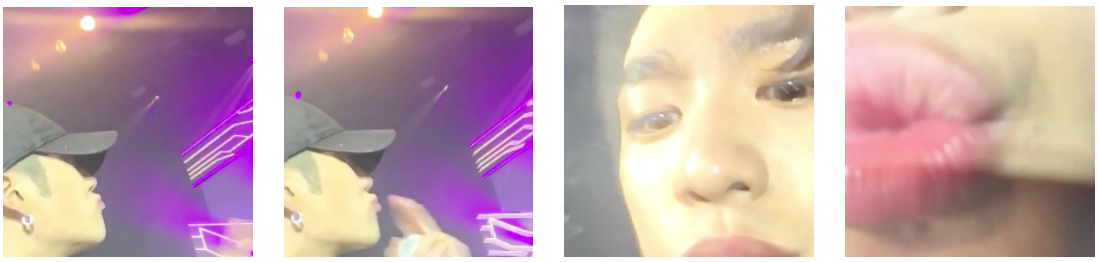
Some data samples of the test dataset and the corresponding generated description are depicted in Figure 5.10. Obvious repetition of relevant n-grams is not happening anymore and the model names subjects slightly more accurately, although it can be noted the redundancy or hallucination of subjects, e.g. *a man* or *a woman*.



(a) CIDEr (b) CIDEr-D

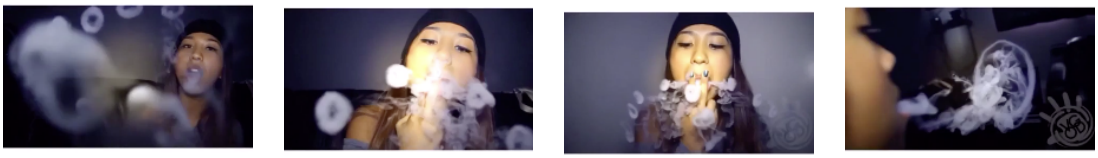
(c) METEOR (d) BLEU-4

Figure 5.6: Metric scores on the test set for a training using self-critic loss with image-only features.



A woman with her hair is looking at a man

A man is dancing in a room with a woman



A young man wearing a hat is dancing in a woman

A woman is sitting at a table with a man

Figure 5.10: Inference of video descriptions for the captioning model using self-critic loss and two sets of ResNet image features. Four frames were sampled from the video uniformly to facilitate understanding for the reader.

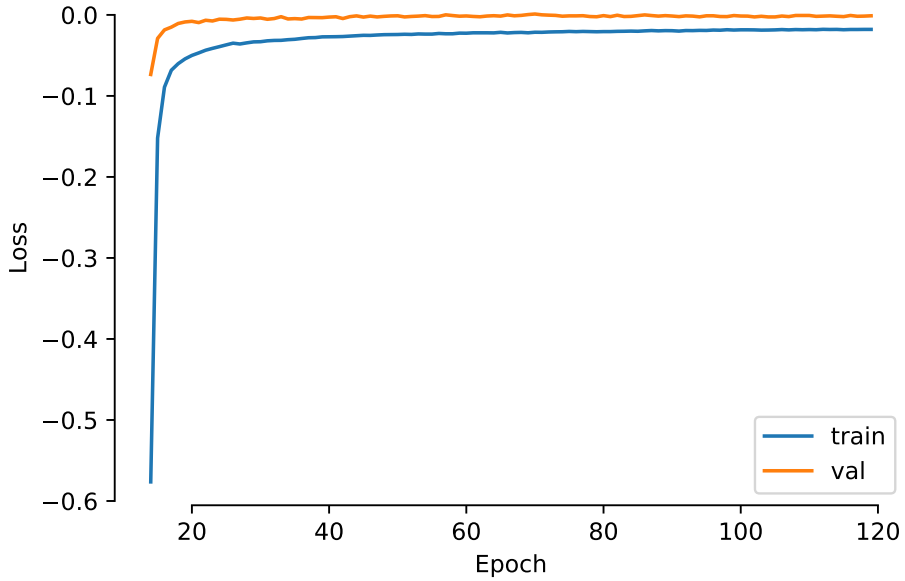


Figure 5.8: Learning curves for a training using self-critic loss and image-only features.

5.5 Experiment 4: Self-critical training with improved features

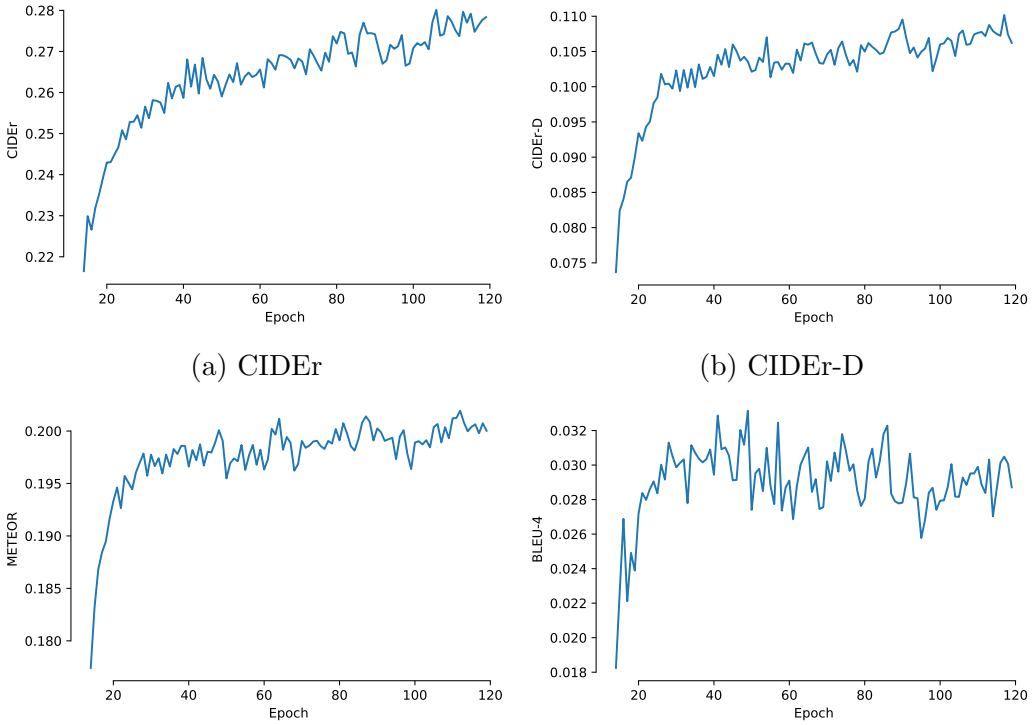
5.5.1 Setup

Once the previous experiment was determined to perform successfully, efforts in improving the model resulted in better input features, called "image+video features" from Section 4.2. Another cross-entropy-trained model had to be trained with these features, as they are still used as initialisation for the self-critical experiments. Apart from this change in the encoder, the rest of the configuration was inherited from the previous experiment.

5.5.2 Results

The progress of the training and validation loss of the model using cross-entropy is displayed in Figure 5.11. The score progression during the training can be also seen in Figure 5.12. Despite little difference can be noticed on the training dynamics, the model takes advantage of the video modality introduced to make a sudden jump in the CIDEr score.

Some data samples of the test dataset and the corresponding generated



(a) CIDEr

(b) CIDEr-D

(c) METEOR

(d) BLEU-4

Figure 5.9: Metric scores on the test set for a training using self-critic loss and image-only features.

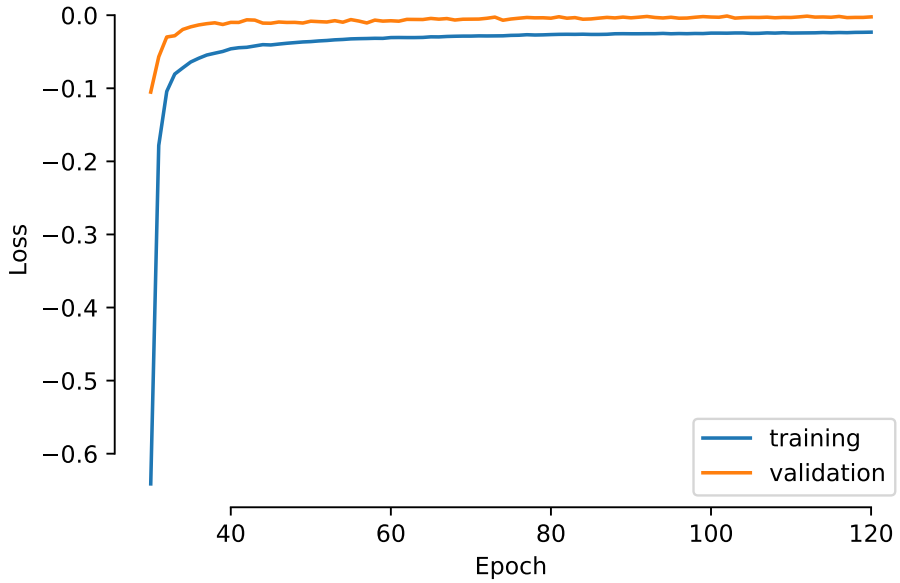


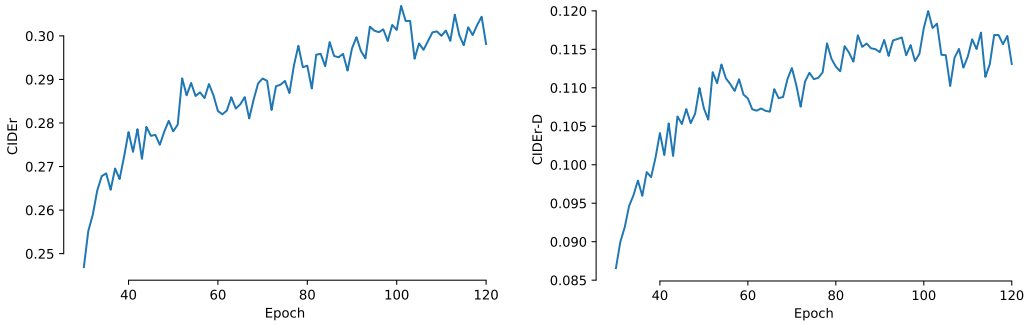
Figure 5.11: Learning curves for a training using self-critic loss with image and video features.

descriptions are depicted in Figure 5.13. The generated descriptions look the same as in the previous experiment, albeit slightly more accurate.

5.6 Quantitative comparison

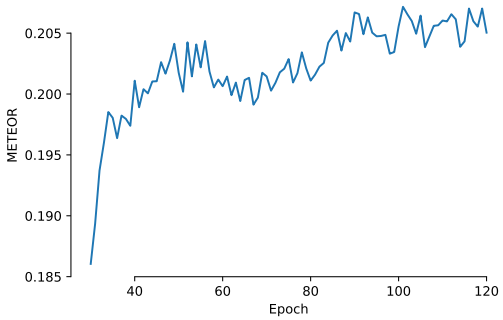
In order to assess the performance of each model and have a global vision of the experiments, a quantitative comparison is first performed by means of automatic NLP metrics. The final scoring for each experiment can be seen in Table 5.4. However, these metrics are not the best indicator of sentence quality (Section 5.8), or at least, they should not be taken into consideration without complementary analyses.

A distribution of the scores for the metrics considered can be found in Figure 5.14. It would be very interesting to have human evaluation in order to discover whether correlations with automatic metrics exist, and to what extent. For BLEU, it is really complicated for models to generate the exact high n-grams as the ground truths, so scores tend to be low and are not sufficiently indicative for model guidance. Regarding CIDEr, scores are slightly higher, as this metric is an averaged measurement of the similarity between generated caption and the ground truths. Therefore it should be relatively more permissive than BLEU, as references for a single sample are

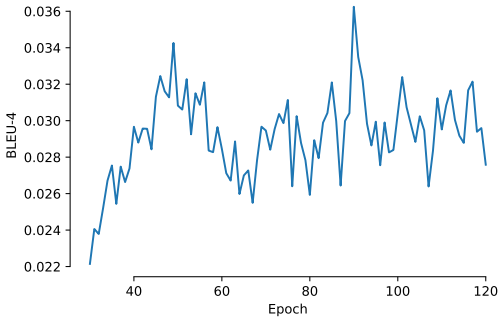


(a) CIDEr

(b) CIDEr-D



(c) METEOR



(d) BLEU-4

Figure 5.12: Metric scores on the test set for a training using self-critic loss with image and video features.

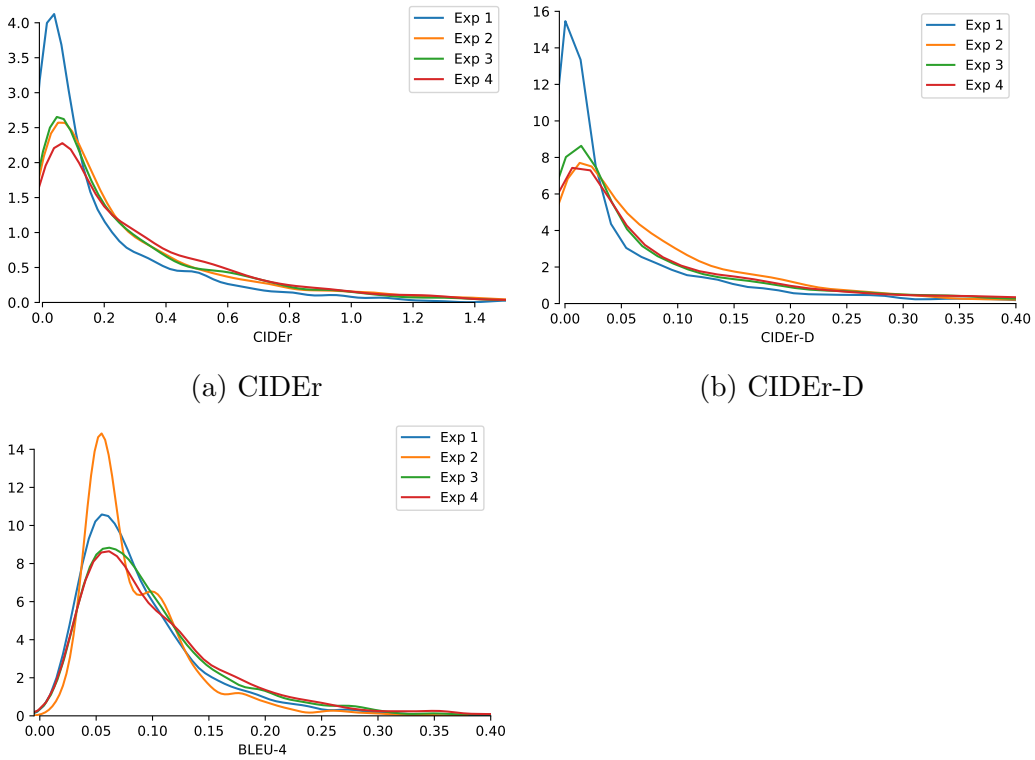
A woman is dancing in a man and a

A young man is eating a plate of food

A man wearing a hat is sitting on a chair

A black cat is playing with a man in a

Figure 5.13: Inference of video descriptions for the captioning model using the better performing ResNet image and I3D video features. Four frames were sampled from the video uniformly to facilitate understanding for the reader.



(a) CIDEr

(b) CIDEr-D

(c) BLEU-4

Figure 5.14: Score distribution for each metric. It was not possible to generate individual scores for METEOR due to technical limitations on the implementation.

roughly consistent. Unfortunately, METEOR score could not be computed individually, but theoretically F_1 score and order penalty over unigrams is less constrained than BLEU and correlates better with human evaluation, so it should be roughly as indicative as CIDEr.

5.7 Qualitative analysis

A more reliable approach for sentence quality assesstment is the direct human comparison of the results to the ground truth. Table 5.5 shows an aggregated qualitative comparison example among the models discussed previously, while the video frames for better reference on how well the results fit are shown in Figure 5.16. Looking at the sentences, we can tell that the generated language of the cross-entropy model is more simple than CIDEr-

Table 5.4: Quantitative comparison of the results. Column *E* is the experiment number. *Feats* are the features used for training, with *image* being traditional and *image+video* being best features, as explained in Section 4.2. *#Ep* is the maximum number of epochs trained, although the best performing is not necessary that number. *C, C-D, M, B-4* are respectively CIDEr, CIDEr-D, METEOR and BLEU-4. One can see that for the model trained with CIDEr, 2 out of 4 scores are better than experiment 3, although this model did not produce any qualitatively good caption due to the metric gaming and closer examination is showed in Section 5.8. Conversely, the model trained with CIDEr-D produced consistent quality captions, which allowed to persist with the training for more epochs.

E	**Feats**	**Reward**	**#Ep**	**C**	**C-D**	**M**	**B-4**
1	image	- (XE)	20	0.1983	0.0737	0.1789	0.0190
2	image	CIDEr	50	**0.2913**	0.0964	**0.2059**	0.0296
3	image	CIDEr-D	120	0.2801	**0.1102**	0.2019	**0.0331**
4	image +video	CIDEr-D	120	**0.3069**	**0.1200**	**0.2072**	**0.0362**

D models, although not necessarily all the time. What is certain is that CIDEr-D models have more tokens matching human descriptions.

Nevertheless, it can be assessed that all the sentences generated by the models are most of the time overly simple, with an elementary description of a subject doing something, sometimes mentioning the place in a plain way. This behaviour can even be noticed with such a simplistic statistic as the length distribution of captions in Figure 5.15. For further analysis, it would be interesting to perform the Turing test on the results, even though most of the results would supposedly not pass it.

Human descriptions are well above complexity levels, with a thorough description of the subject, naming a number of details. The sentences include more than one verb, describing the action through time. This last point is very interesting, as all learned models, even the ones using video features, give the impression that they are describing static scenes, like a photograph taken from the video. Lastly, human descriptions are occasionally supplemented with quotations from the audio conversations or text shown on signs during the video, which enriches or provides more context to the captions.

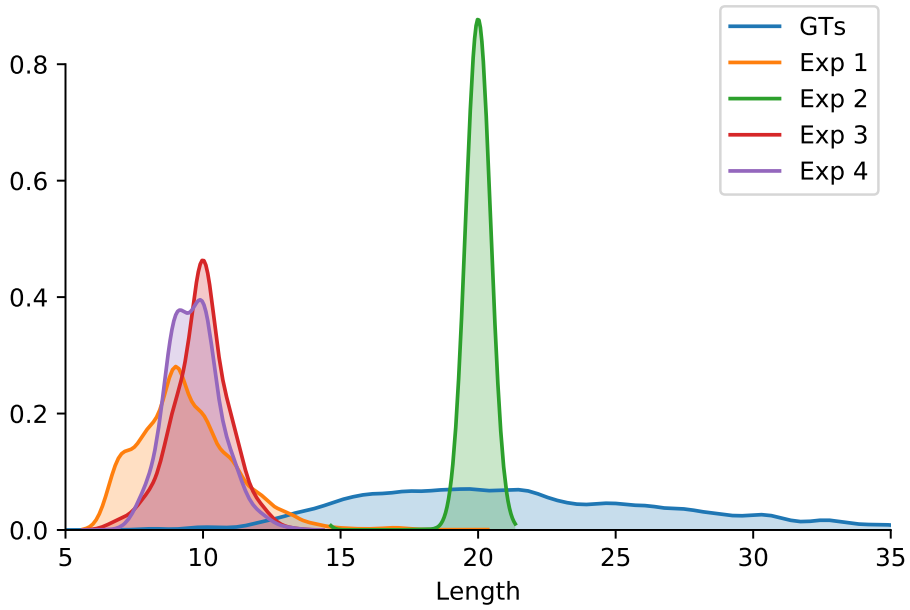


Figure 5.15: Distribution of the description lengths. *GTs* is the averaged length of each ground truth set. Most models produce very short sentences, while natural descriptions have a rich distribution with a long right tail. The maximum sequence length is set to 20, therefore the second experiment output has most of its distribution mass there in order to abuse the metric as much as possible.

Table 5.5: Qualitative comparison of model performance over samples. *Ref 1* is the first reference for that sample. *Exp x* is the output from the experiments presented. Video frames are shown in Figure 5.16.

Ref 1	*Man in sports uniform jacket pounds one fist into his palm in front of a microphone*
Exp 1	*A man in a black jacket is smiling*
Exp 2	*A young man wearing a man in a man is talking and a man with a man in a man*
Exp 3	*A young man wearing a black shirt is talking and smiling*
Exp 4	*A man wearing a black shirt is talking into a microphone*
Ref 1	*Black and white dog runs around a track with a kerchiefed monkey on its back.*
Exp 1	*A man is riding a horse and then jumps over a fence*
Exp 2	*A man riding a horse on a horse is riding a horse in a horse is riding a horse in*
Exp 3	*A man riding a horse in a woman*
Exp 4	*A man is riding a motorcycle on a track*
Ref 1	*Soldiers stand at ease in formation as a girl in animal skin dress rushes into their midst to jump on and wrap her legs around a man in the second row, and other women seek their loved ones.*
Exp 1	*A woman is holding a microphone and dancing*
Exp 2	*A man and a woman are dancing in a man and a woman in a man is a woman in*
Exp 3	*A man and a woman are dancing in a man*
Exp 4	*A man and a woman are walking in a man*
Ref 1	*Man sitting, puts gold colored scarf around neck, picks up a leather briefcase, puts it on his lap.*
Exp 1	*A man is sitting on a couch and talking*
Exp 2	*A man is sitting on a man is sitting on a man is a man in a man is sitting*
Exp 3	*A man is sitting on a chair and a woman*
Exp 4	*A man is sitting on a chair and a woman*

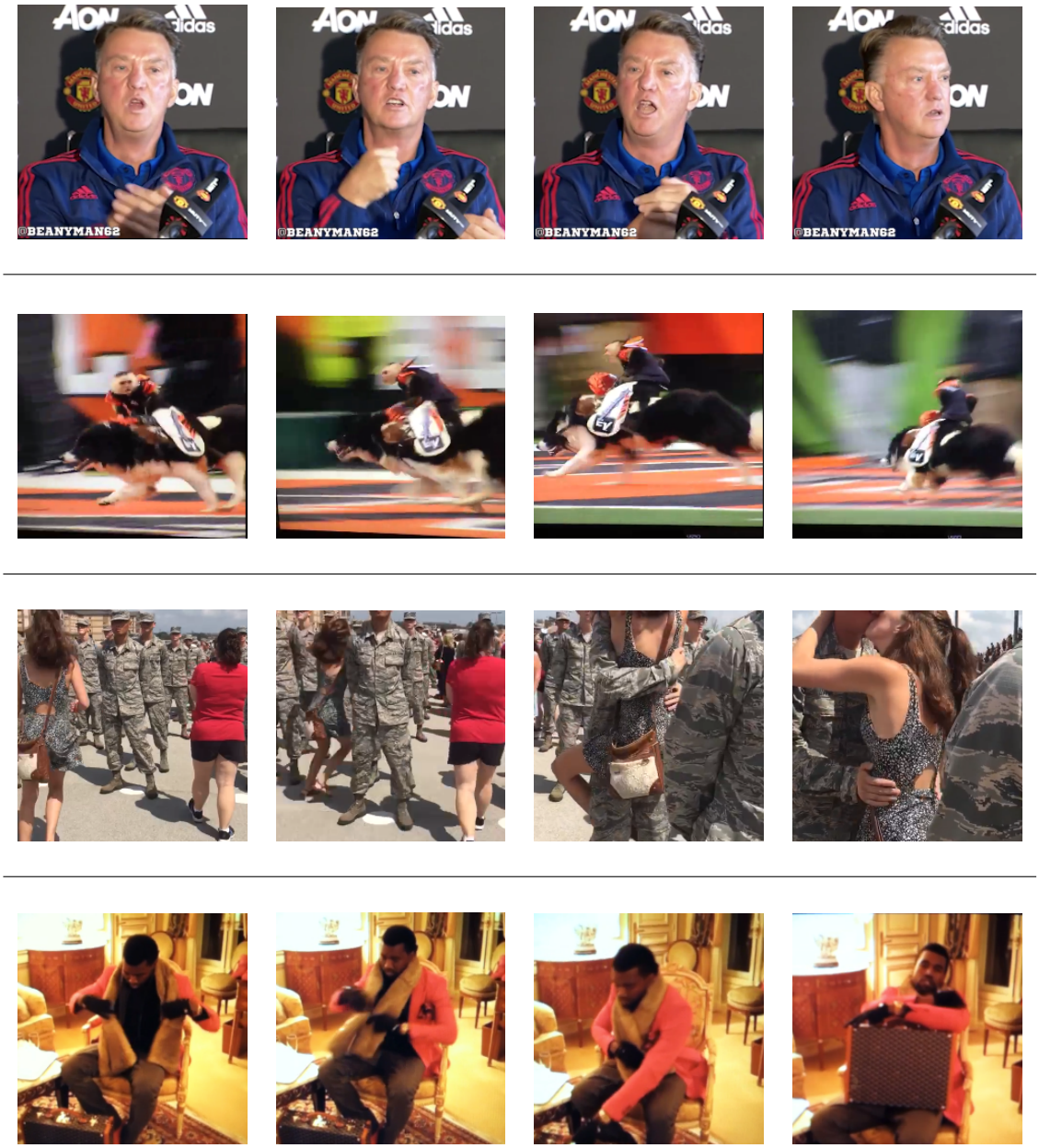


Figure 5.16: Frames corresponding to the description comparison from Table 5.5, respectively.

5.8 Study of metric abuse by the system

This section shows how models can abuse non-robust NLP metrics, achieving high scores without actually exhibiting the complexity required for those scores. This is called abusing or "gaming" the metric, or simply exploiting its design flaws. When susceptible metrics are used for methods like self-critical training, degenerate cases can occur as seen in Section 5.3. The metric selected for this study is BLEU, although it acts like a proxy for any arbitrary non-robust metric.

Let us first quote BLEU equation (3.14) and equation (3.16) from Section 3.5.1 to understand how this metric can be abused:

$$\text{BLEU-4} = BP \cdot \left(\prod_{n=1}^{4} p_n \right)^{\frac{1}{4}} \tag{5.1}$$

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases} . \tag{5.2}$$

A brevity penalty BP can be seen as a gate that opens as the length of the references increases, it starts from zero and there is a cap on one. It is worth noting that the reference length taken is the shortest one from the reference set. The penalty can be heavily abused by arbitrarily repeating words until a favourable minimum is reached.

What follows is the geometric average of n-gram precisions p_n, which can be again tricked using repetition. In this case, repeating lower n-grams, as greater number of higher n-grams would require too much complexity. This can become even worse when a smoothing function is used, because the number of any n-gram overlapping (presumably higher ones under gaming circumstances) can reach zero. The rationale can be verified in Table 5.6, in which each piece of the metric is computed separately.

A model trained to game a metric will therefore produce a short sentence with a high number of repetitions. It can score very high if those instances of words appear in the reference text, but could also fail miserably if there are no higher n-grams generated that match with a reference. This is obviously due to the multiplication on the geometric average, which will deteriorate all other abused scores. We can then state that a model that fails to understand a scene will produce noisy scores with either very high or very low values.

Some negative examples can be seen in Table 5.7, and positive in Table 5.8. These samples are from the same models, *Model 1* is the cross-entropy model trained with image-only features, *Model 2* is the self-critical model trained with image-only features. For the last model, the CIDEr score

Table 5.6: Demonstration of BLEU-4 abuse. *BP* stands for the brevity penalty, *n-gram* is the *n*-gram precision, *Prec.* is the geometric averaged precision and *Score* is the final score obtained by multiplying the brevity penalty with the geometric average. The first candidate is a regular generated caption. The second one shows 1-gram repetition. The last one shows $2-3$-gram repetitions.

References
At an outside sporting event, a blue shirted, red-headed man jumps on the back of a horse with a brown haired person in front as the head of the horse.
A man in a blue shirt on a track field outside jumps onto a fake horse with another man wearing a gold shirt.
At a track a blond man does a jig and then runs and jumps onto the back of a person that is the rear end of a human horse.
An athlete with orange hair in blue and white on a red field with white stripes, jumps on the back of two people dressed as a horse.
Two asian men, acting as mascots for a track and field team, perform antics on the field.

Candidates						
BP	**1-gram**	**2-gram**	**3-gram**	**4-gram**	**Prec.**	**Score**
A wrestler is <unk>another wrestler down the mat						
0.4421	47/49	34/48	19/47	11/46	0.5062	0.2238
a a a a a a a a a a a a a a a a a a a a a man jumps on a						
0.5860	47/58	25/57	18/56	12/55	0.3973	0.2328
A basketball player is a man is a man is a man is a woman is a man is						
0.7484	67/69	41/68	19/67	7/66	0.3643	0.2726

Table 5.7: Samples where a model abusing BLEU-4 metric achieves lower score than the model it used for initialisation.

Model	Score	Diff	Caption
1	0.2812		*A person is jumping a skateboard over a ramp*
2	0.0781	-0.2031	*A woman is in a bathroom with a mirror and a woman in a bathroom with a woman in*
1	0.2665		*A young man is smiling and laughing*
2	0.0567	-0.2098	*A woman with long hair is a girl with a woman is smiling and a woman is her hair*
1	0.3005		*A cat is wearing a santa hat*
2	0.0757	-0.2247	*A young boy wearing a cat is wearing a cat is wearing a cat is wearing a cat is*

Table 5.8: Samples where a model abusing BLEU-4 metric achieves higher score than the model it used for initialisation.

Model	Score	Diff	Caption
1	0.0692		*A man is sitting in a chair and making faces*
2	0.2730	+0.2038	*A young man with a man is wearing a red shirt is a man is sitting on a man*
1	0.0492		*A woman is sitting on a couch and talking to a man*
2	0.2566	+0.2073	*Two girls are sitting on a bench with a woman sitting on a bench and a woman is a*
1	0.0699		*A dog is jumping up and down on a bed*
2	0.2985	+0.2286	*A black and white dog is a dog is playing with a dog is a dog is a dog*

is better than the first model 72.41% of the time, while for BLEU the percentage goes down to 50.92%.

5.9 No stopping criteria

This section discusses a notorious drawback on self-critical approaches, which is the lack of an indicator of early-stopping, or signs of qualitative detriment on the model performance during training.

Learning curves of self-critical training can be seen in Figure 5.17. It is visible that the training and validation losses approach zero during the training while we are trying to minimise the loss function. At the first glance,

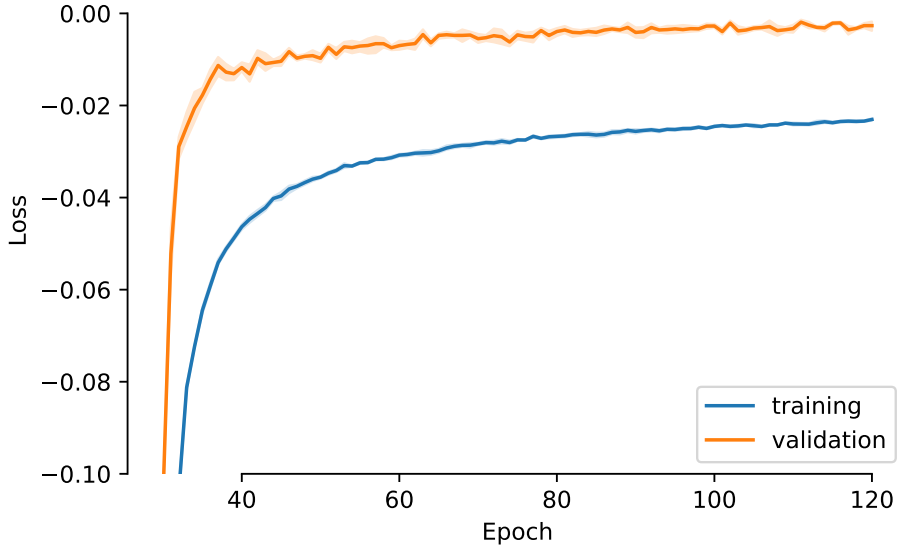
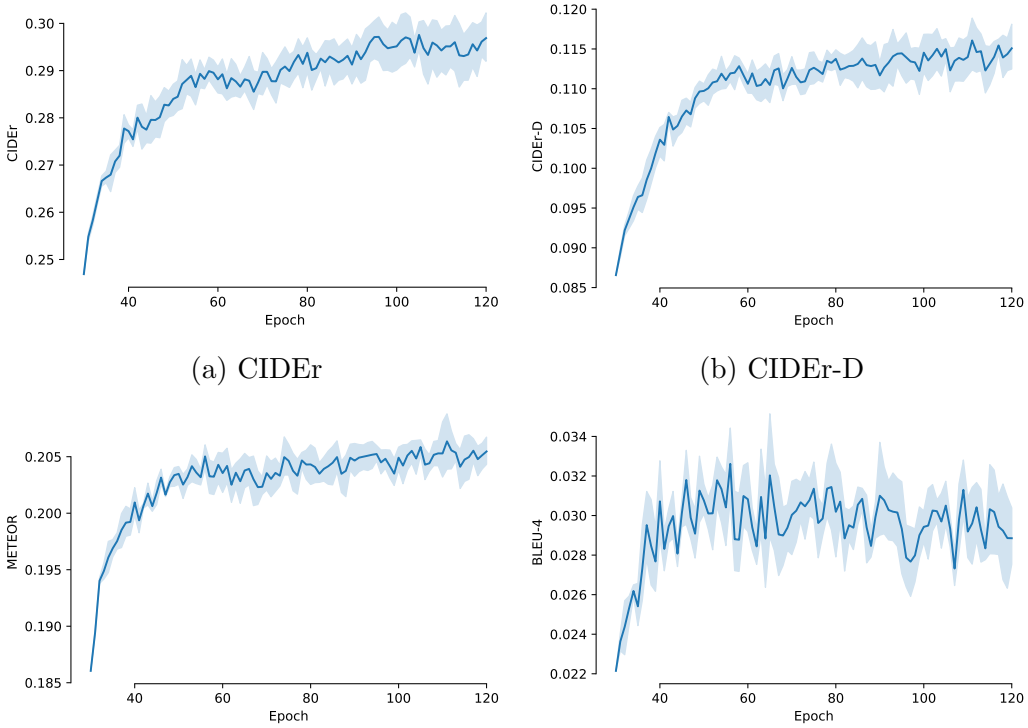


Figure 5.17: Learning curves of a model trained with self-critic loss. It may be worth noting that the validation loss reaches zero faster than the training loss, contrary to traditional ML. This is due to the greedy sampling performed during inference, while in the training stage a stochastic sampling mechanism is used to compute the reward difference (Section 3.3.2).

it may seem strange that the validation loss reaches zero to continue oscillating around, if we consider that it should eventually overfit. However, in Reinforcement Learning, overfitting is not spotted the same way as in cross-validation learning, so it renders this plot slightly more uninformative.

Turning to metric scores does not offer a better vantage. Figure 5.18 shows ever-improving scores, making it difficult to draw clear conclusions. For these reasons, the stopping criteria had to be addressed manually. The model was allowed to train for a number of epochs, until the quality of the results started decreasing. For a metric that allowed gaming this point was below 50 epochs, while for others better designed such as CIDEr-D, ranges of 150 epochs were allowed.



(a) CIDEr

(b) CIDEr-D

(c) METEOR

(d) BLEU-4

Figure 5.18: Metric scores on the test set. Five independent training runs on the fourth experiment were performed.

Chapter 6

Discussion

This chapter discusses the results and analyses of the previous chapter about using a reinforcement learning method like self-critic for sequence learning. It explains the behaviour of the models and their reason for it, discussing the use of automatic metrics in reward-based training. It also mentions the importance of rich and diverse data and finishes discussing the language model.

As stated in the comparison with human-made descriptions in Section 5.7, the current models lack the inherent complexity to produce a complete, multi-faceted caption of a video. These models seem to be driven to name only objects and attributes that they are very confident about, followed by an action over one object at most. The descriptions are rigid and never change their structure or flow of the sentence, albeit the references exhibit a different number of these changes. An example of this is the last sample from Table 5.5, "*Man sitting, puts gold colored scarf around neck, picks up a leather briefcase, puts it on his lap*". When the scene is complex enough, a succession of different actions takes place on human-made descriptions. Conversely, the model tries to summarise the whole video with the same descriptive structure unsuccessfully, while additionally trying to output high-frequency tokens as much as the metric allows in order to score better.

Throughout the experiments, it has been shown that automatic metrics should not be trusted anymore for quality assessment or any major criteria, e.g. when to cease the training, due to the inherent ability of optimisation mechanisms to find flaws in human-designed measures. A second important aspect to notice is the only minor increase in overall performance from an improved encoding, which mainly suggests that the performance bottleneck is in the decoding submodule. Finally, it is also appreciated how the careful choice of a well-crafted metric is a decisive factor not only in the scoring stage, but also in shaping the model behaviour in the learning stage.

Concretely, the short descriptions seem to be produced by the narrow search space of tokens that the model considers plausible to output, which leads to a lack of diversity in the generated captions for different videos, e.g. collapsing on the same phrase for close enough scenes or inability to use synonyms. These problems lead to absence of context ("*a man is smiling to a woman*"), which is undesirable for high-quality descriptions. These issues about training with scoring functions as reward have been previously discussed [62], reaching a further conclusion that automatic metrics are not reliable for either training or evaluation, and suggesting that learnable rewards should be used instead for the case of Reinforcement Learning-based training for captioning.

Regarding the data to be used to train a model with, it was discussed in Section 3.5.3 that human assessment and human description is a diverse and multi-perspective process, so it cannot be reduced to a single golden reference or perspective. Due to this, datasets should have more than one sentence per sample as (1) metrics work better when there are more than one reference, and (2) scenes can be described similarly with no matching words, which should be reflected in the dataset for the model to learn. Failure to provide diverse samples will hurt the training and skew the model towards certain ways of description and certain tokens, eventually hurting the overall performance as the model will keep repeating those biased words or tokens that it is most over-confident about for descriptions and contexts that are not a good match.

Taking into consideration the architecture in addition to the loss function, the baseline model trained with cross-entropy is already displaying this short-sighted, rigid descriptive behaviour. There exists the possibility that the problem lays in the language model, that may have too low complexity in order to present different, adaptive behaviours conditioned on the input. Given a model with the right capacity, maybe the self-critic loss could train it to the level to be as nuanced as human descriptions. However, the answer is unknown and more experiments with language models of different nature and complexity should be made.

Chapter 7

Conclusions

This thesis has provided an experimental analysis on different training mechanisms for language generation tasks using the popular encoder-decoder architecture. The model was trained using well-known datasets with a great quantity and variety of samples and human descriptions, and image and video features were extracted from these datasets for the experiments.

A baseline model with cross-entropy loss was first trained to serve additionally as initialisation, producing average-quality descriptions. Further self-critical models trained with a regular metric as reward produced heavily distorted captions, so not every function is suitable. When scoring functions robust against gaming are used, the model cannot abuse them and it is able to outperform the baseline. However, the experiment results did not reflect a huge improvement on the quality. Neither did they present complex, unseen behaviour, but rather the expressiveness worsened and the vocabulary moved closer to what the training reference uses.

With the use of reinforcement learning approaches, the model was driven to learn a more accurate behaviour of what is expected to output. Results showed that this can be used as a satisfactory correcting method and consequently a metric score booster for an already sound model. Unfortunately, the policy that can be refined was shown to be heavily constrained by the agent capabilities, i.e. architectural design, mainly the language model.

The last experiment showed that greatly improving the encoding mechanism with the addition of a second video modality, which regards images over time, resulted in just a marginal enhancement of the overall results. This suggests again that the decoder, the second component constituted by a language model, is the one holding down substantial performance gains. Therefore, this new learning technique cleverly solves certain problems from the traditional training, although to assess its real potential, further analysis with higher capacity language models needs to be conducted.

7.1 Future work

A number of methods can be applied to improve the model performance. Unfortunately, time constraints did not allow further progress, but all of them look promising in this joint area of image captioning and reinforcement learning.

The most obvious next step could be to augment the decoder with attention mechanisms [5] to assist the decoder memory to deal with longer dependencies. This has been reported as an improvement even in this same task by other groups [21]. Continuing with decoding, it would be interesting to use a Transformer-based decoder [57], not recurrent by nature, to test if it helps to the current short length of the generated captions.

It is worth noting that no drastic tuning of hyperparameters has been performed. A sound amendment is the use of Cyclical Learning Rates [53], which is a learning rate schedule that enables the model to (1) step out sharp local minima, and (2) traverse plateau regions faster while on high learning rates. A follow-up work that can be combined with this technique is Stochastic Weight Averaging [23], which has been reported [41] as a performance improvement for policy gradient methods and uses averages of several SGD-trained solutions.

Given the shortcomings of automatic metrics, interesting recent work on visual storytelling [62] addresses in a sensible way that can be translated to image captioning. Instead of using an intrinsically flawed automatic metric, a reward function is learned. This work reports more human-like results compared to any specific metric used. Another beneficial addition is the use of Beam Search on the decoding stage. Concretely, Diverse Beam Search [59] holds all the advantages of the vanilla version while fostering diversity on the search process.

Finally, an improvement not for the Reinforcement Learning training, but for the previous Cross-Entropy one can be made by using a modified loss function named Frequency Aware Cross-Entropy [24]. This loss penalises the use of high-frequency words in order to improve the diversity of the generated sentences. This would provide a better starting point for the second stage of the training, and promote exploration due to a more balanced weight distribution over tokens.

Bibliography

[1] Akutsu, T., Kuhara, S., Maruyama, O., and Miyano, S. A system for identifying genetic networks from gene expression patterns produced by gene disruptions and overexpressions. *Genome Informatics 9* (1998), 151–160. 11

[2] Awad, G., Butt, A., Curtis, K., Lee, Y., Fiscus, J., Godil, A., Joy, D., Delgado, A., Smeaton, A. F., Graham, Y., Kraaij, W., Quénot, G., Magalhaes, J., Semedo, D., and Blasi, S. Trecvid 2018: Benchmarking video activity detection, video captioning and matching, video storytelling linking and video search. In *Proceedings of TRECVID 2018* (2018), NIST, USA. 22

[3] Ba, J., Mnih, V., and Kavukcuoglu, K. Multiple object recognition with visual attention. In Bengio and LeCun [7]. 13

[4] Bahdanau, D., Brakel, P., Xu, K., Goyal, A., Lowe, R., Pineau, J., Courville, A. C., and Bengio, Y. An Actor-Critic Algorithm for Sequence Prediction. *CoRR abs/1607.07086* (2016). 17

[5] Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* (2015). 13, 70

[6] Bengio, S., Vinyals, O., Jaitly, N., and Shazeer, N. Scheduled sampling for sequence prediction with recurrent neural networks. *CoRR abs/1506.03099* (2015). 8, 17, 29

[7] Bengio, Y., and LeCun, Y., Eds. *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* (2015). 71, 73, 74

[8] Buchanan, B. G., and Duda, R. O. Principles of rule-based expert systems. In *Advances in computers*, vol. 22. Elsevier, 1983, pp. 163–216. 11

[9] Buhrmester, M., Kwang, T., and Gosling, S. D. Amazon's mechanical turk: A new source of inexpensive, yet high-quality, data? *Perspectives on Psychological Science 6*, 1 (2011), 3–5. PMID: 26162106. 7, 19

[10] Carreira, J., and Zisserman, A. Quo vadis, action recognition? A new model and the kinetics dataset. *CoRR abs/1705.07750* (2017). 33, 34

[11] Chen, X., Fang, H., Lin, T., Vedantam, R., Gupta, S., Dollár, P., and Zitnick, C. L. Microsoft COCO captions: Data collection and evaluation server. *CoRR abs/1504.00325* (2015). 7, 19

[12] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014). 8, 13

[13] Donahue, J., Hendricks, L. A., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., and Darrell, T. Long-term recurrent convolutional networks for visual recognition and description. *CoRR abs/1411.4389* (2014). 12

[14] Farhadi, A., Hejrati, M., Sadeghi, M. A., Young, P., Rashtchian, C., Hockenmaier, J., and Forsyth, D. Every picture tells a story: Generating sentences from images. In *European conference on computer vision* (2010), Springer, pp. 15–29. 11

[15] Feichtenhofer, C., Pinz, A., and Zisserman, A. Convolutional two-stream network fusion for video action recognition. *CoRR abs/1604.06573* (2016). 33

[16] Gerber, R., and Nagel, N. . Knowledge representation for the generation of quantified natural language descriptions of vehicle traffic in image sequences. In *Proceedings of 3rd IEEE International Conference on Image Processing* (Sep. 1996), vol. 2, pp. 805–808 vol.2. 11

[17] group Aalto University, C. Deepcaption. `https://github.com/aalto-cbir/DeepCaption`, 2019. 40

[18] He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. *CoRR abs/1512.03385* (2015). 33

[19] Hochreiter, S., and Schmidhuber, J. Long short-term memory. *Neural Computation 9*, 8 (Nov 1997), 1735–1780. 27

[20] Hornik, K., Stinchcombe, M., and White, H. Multilayer feedforward networks are universal approximators. *Neural networks 2*, 5 (1989), 359–366. 32

[21] Huang, P.-Y., Liang, J., Vaibhav, V., Chang, X., and Hauptmann, A. Informedia@ TRECVID 2018: Ad-hoc video search with discrete and continuous representations. *TRECVID Proceedings* (2018). 70

[22] Ioffe, S., and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR abs/1502.03167* (2015). 33

[23] Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D. P., and Wilson, A. G. Averaging weights leads to wider optima and better generalization. *CoRR abs/1803.05407* (2018). 70

[24] Jiang, S., Ren, P., Monz, C., and de Rijke, M. Improving neural response diversity with frequency-aware cross-entropy loss. *CoRR abs/1902.09191* (2019). 70

[25] Karpathy, A., and Li, F. Deep visual-semantic alignments for generating image descriptions. *CoRR abs/1412.2306* (2014). 13, 14

[26] Keneshloo, Y., Shi, T., Ramakrishnan, N., and Reddy, C. K. Deep reinforcement learning for sequence to sequence models. *CoRR abs/1805.09461* (2018). 30

[27] Kingma, D. P., and Ba, J. Adam: A method for stochastic optimization. In Bengio and LeCun [7]. 44

[28] Kiros, R., Salakhutdinov, R., and Zemel, R. Multimodal neural language models. In *International Conference on Machine Learning* (2014), pp. 595–603. 12

[29] Kiros, R., Salakhutdinov, R., and Zemel, R. S. Unifying visual-semantic embeddings with multimodal neural language models. *CoRR abs/1411.2539* (2014). 13

[30] Krizhevsky, A., Sutskever, I., and Hinton, G. E. ImageNet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (2012), pp. 1097–1105. 8

[31] Krogh, A., and Hertz, J. A. A simple weight decay can improve generalization. In *Advances in neural information processing systems* (1992), pp. 950–957. 42

[32] Kulkarni, G., Premraj, V., Ordonez, V., Dhar, S., Li, S., Choi, Y., Berg, A. C., and Berg, T. L. Babytalk: Understanding and generating simple image descriptions. *IEEE Transactions on Pattern Analysis and Machine Intelligence 35*, 12 (2013), 2891–2903. 11

[33] Lamb, A. M., Goyal, A. G. A. P., Zhang, Y., Zhang, S., Courville, A. C., and Bengio, Y. Professor forcing: A new algorithm for training recurrent networks. In *Advances In Neural Information Processing Systems* (2016), pp. 4601–4609. 8

[34] Lavie, A., and Denkowski, M. J. The meteor metric for automatic evaluation of machine translation. *Machine Translation 23*, 2-3 (Sept. 2009), 105–115. 35

[35] Li, S., Kulkarni, G., Berg, T. L., Berg, A. C., and Choi, Y. Composing simple image descriptions using web-scale n-grams. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning* (2011), Association for Computational Linguistics, pp. 220–228. 11

[36] Li, Y., Song, Y., Cao, L., Tetreault, J. R., Goldberg, L., Jaimes, A., and Luo, J. TGIF: A new dataset and benchmark on animated GIF description. *CoRR abs/1604.02748* (2016). 21

[37] Lin, T., Maire, M., Belongie, S. J., Bourdev, L. D., Girshick, R. B., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft COCO: common objects in context. *CoRR abs/1405.0312* (2014). 19

[38] Mao, J., Xu, W., Yang, Y., Wang, J., and Yuille, A. L. Deep captioning with multimodal recurrent neural networks (m-rnn). In Bengio and LeCun [7]. 12

[39] Miller, G. A. Wordnet: A lexical database for english. *Commun. ACM 38*, 11 (Nov. 1995), 39–41. 35

[40] Mnih, A., and Hinton, G. Three new graphical models for statistical language modelling. In *Proceedings of the 24th international conference on Machine learning* (2007), ACM, pp. 641–648. 12

[41] Nikishin, E., Izmailov, P., Athiwaratkun, B., Podoprikhin, D., Garipov, T., Shvechikov, P., Vetrov, D., and Wilson, A. G. Improving stability in deep reinforcement learning with weight averaging. In *Artificial Intelligence Workshop on Uncertainty in Deep Learning* (2018). 70

[42] Olah, C. Understanding LSTM Networks. 27, 28

[43] Papineni, K., Roukos, S., Ward, T., and jing Zhu, W. Bleu: a method for automatic evaluation of machine translation. pp. 311–318. 8, 34

[44] Pascanu, R., Mikolov, T., and Bengio, Y. Understanding the exploding gradient problem. *CoRR abs/1211.5063* (2012). 42

[45] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop* (2017). 40

[46] Porter, M. F. Readings in information retrieval. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997, ch. An Algorithm for Suffix Stripping, pp. 313–316. 35

[47] Ranzato, M., Chopra, S., Auli, M., and Zaremba, W. Sequence level training with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings* (2016), Y. Bengio and Y. LeCun, Eds. 8, 17, 32

[48] Rennie, S. J., Marcheret, E., Mroueh, Y., Ross, J., and Goel, V. Self-critical sequence training for image captioning. *CoRR abs/1612.00563* (2016). 8, 17, 18, 32, 42, 43

[49] Robertson, S. Understanding inverse document frequency: On theoretical arguments for IDF. *Journal of Documentation 60* (2004), 2004. 36

[50] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein,

M. S., Berg, A. C., and Li, F. ImageNet large scale visual recognition challenge. *CoRR abs/1409.0575* (2014). 8

[51] Sharma, P., Ding, N., Goodman, S., and Soricut, R. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pp. 2556–2565. 15, 16

[52] Sjöberg, M., Tavakoli, H. R., Xu, Z., Mantecón, H. L., and Laaksonen, J. PicSOM Experiments in TRECVID 2018 Workshop notebook paper. 38, 39, 43

[53] Smith, L. N. No more pesky learning rate guessing games. *CoRR abs/1506.01186* (2015). 70

[54] Sutton, R. S., and Barto, A. G. Reinforcement learning: An introduction. 17

[55] Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., and Liu, C. A survey on deep transfer learning. *CoRR abs/1808.01974* (2018). 32

[56] Turian, J. P., Ratinov, L.-A., and Bengio, Y. Word representations: A simple and general method for semi-supervised learning. In *ACL* (2010). 28

[57] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. *CoRR abs/1706.03762* (2017). 15, 16, 70

[58] Vedantam, R., Lawrence Zitnick, C., and Parikh, D. Cider: Consensus-based image description evaluation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2015). 8, 36

[59] Vijayakumar, A. K., Cogswell, M., Selvaraju, R. R., Sun, Q., Lee, S., Crandall, D. J., and Batra, D. Diverse beam search: Decoding diverse solutions from neural sequence models. *CoRR abs/1610.02424* (2016). 70

[60] Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. Show and tell: A neural image caption generator. *CoRR abs/1411.4555* (2014). 8, 12, 28, 29

[61] Wang, C., Yang, H., Bartz, C., and Meinel, C. Image captioning with deep bidirectional LSTMs. *CoRR abs/1604.00790* (2016). 13, 14

[62] Wang, X., Chen, W., Wang, Y., and Wang, W. Y. No metrics are perfect: Adversarial reward learning for visual storytelling. *CoRR abs/1804.09160* (2018). 68, 70

[63] Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning 8*, 3 (May 1992), 229–256. 8, 17, 31

[64] Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A. C., Salakhutdinov, R., Zemel, R. S., and Bengio, Y. Show, attend and tell: Neural image caption generation with visual attention. *CoRR abs/1502.03044* (2015). 13, 15

[65] Yao, B. Z., Yang, X., Lin, L., Lee, M. W., and Zhu, S. I2t: Image parsing to text description. *Proceedings of the IEEE 98*, 8 (Aug 2010), 1485–1508. 11

[66] Young, T., Hazarika, D., Poria, S., and Cambria, E. Recent trends in deep learning based natural language processing. *CoRR abs/1708.02709* (2017). 27

[67] Zaremba, W., and Sutskever, I. Reinforcement learning neural turing machines. *CoRR abs/1505.00521* (2015). 31